Trabajo Fin de Grado

Diseño, construcción, sensorización y control de un robot imprimible de exteriores

Daniel González Arribas
Tutor: Alberto Valero Gómez

Departamento de Ingeniería de Sistemas y Automática

Agradecimientos

Quiero agradecer a Juan González Gómez y a Alberto Valero su excepcional trabajo promoviendo la adopción de las impresoras 3D en la Universidad Carlos III de Madrid y el movimiento de los robots imprimibles, ya que sin ambas iniciativas este trabajo nunca hubiera existido: su entusiasmo y esfuerzo, así como el de todos los colaboradores en estos proyectos, es lo que ha hecho posible. Quiero agradecer también a Alberto Valero su excelente labor como tutor en la elaboración de este trabajo.

Agradezco también a Marco Esteban Illescas su colaboración en la fabricación de las piezas del prototipo final.

Por último, quiero agradecer a mi familia su valioso apoyo durante todos estos meses de trabajo.

Resumen

El objetivo de este proyecto es la creación de un robot imprimible basado en tracks articulados. El robot se desarrollará con tecnologías libres y la estructura se creará mediante una impresora 3D de tipo RepRap. Se creará el modelo cinemático del robot y se analizarán matemáticamente las restricciones de movimiento. Se dotará al robot de una sistema de percepción basado en dos sensores de proximidad basados en infrarrojos y un acelerómetro analógico. Se diseñará un sistema electrónico de control basado en un microcontrolador Arduino Mega2560 comunicado con un ordenador personal. Se programará un software de control mediante el PC y se programarán varias maniobras autónomas.

Abstract

The main goal of this project is to design and build a printable mobile robot with track-based motion. The vehicle will be designed making use of several open source technologies and the physical structure of the robot will be manufactured on a RepRap 3D printer. A kinematic model of the robot will be developed and the motion constraints will be mathematically analyzed. The robot will be fitted with a sensorial system composed by two IR-based proximity sensors and an analog accelerometer. An electronic control system will be designed on top of an Arduino Mega2560 microcontroller board connected to a personal computer. Finally, the microcontroller will be programmed to provide both remote control functionality and some autonomous capabilities.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El propósito de este capítulo es presentar el proyecto. Se enumeran los objetivos del proyecto, se explica la motivación, se presenta el concepto de robots imprimibles y se explican brevemente las tecnologías empleadas.

El capítulo está compuesto por las siguientes secciones:

- La sección 1.1 expone los objetivos del proyecto.
- La sección 1.2 explica la motivación y los antecedentes del trabajo.
- La sección 1.3 enumera las fases del proyecto.
- La sección 1.4 introduce las tecnologías empleadas.

1.1. Objetivos del proyecto

Se han especificado los siguientes objetivos del proyecto:

- Desarrollar un robot móvil imprimible basado en "tracks" u orugas móviles capaz de moverse en terrenos complicados y de sortear diversos tipos de obstáculos. El robot presentará una morfología similar a su predecesor, el F-Track (presentado en la sección 1.2), pero debe implementar mejoras mecánicas que solucionen los problemas mecánicos de este.
- Diseñar el robot de modo que sea fácilmente reproducible y modificable, de acuerdo a la filosofía de los robots imprimibles. Por ello, se debe favorecer el uso de tecnologías libres (u "open-source") y accesibles durante el desarrollo. Se crearán las piezas del robot mediante una impresora 3D de tipo RepRap y se caracterizará una metodología de trabajo para el desarrollo de robots imprimibles basados en esta tecnología de fabricación
- Modelar matemáticamente el robot y crear modelos cinemáticos simplificados para las situaciones de operación. Emplear la teoría de mecanismos para determinar las relaciones que deben cumplir las velocidades de los motores que desplazan al robot en función de la velocidad global del robot, el relieve y de la posición de las articulaciones.
- Crear un sistema de percepción básico para el robot, de modo que pueda detectar obstáculos y características del terreno, así como calcular su inclinación.
- Programar un software de control que demuestre la funcionalidad del robot y permita comunicarlo con un ordenador personal de forma inalámbrica.
- Demostrar la viabilidad de la robótica imprimible de bajo coste y sus ventajas.

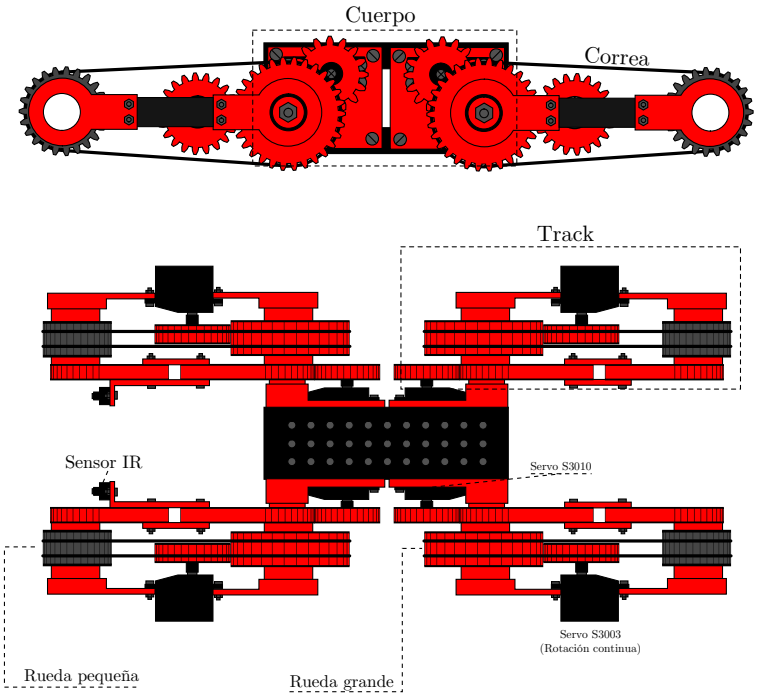

Figura 1.1: Diagrama del D-Track, el robot desarrollado en este proyecto

1.2. Motivación y antecedentes

1.2.1. Robots imprimibles

El avance de la impresión 3D accesible de bajo coste durante los últimos años gracias al proyecto RepRap (véase la sección 1.4) ha posibilitado la aparición de plataformas de robótica open-source en el ámbito académico. Las principales ventajas de estas plataformas son, de acuerdo a González Gómez et al [1]:

- Prototipado rápido de robots
- Bajo coste de impresión
- Facilidad de adaptación y reconfiguración (evolución)
- Facilidad de compartición de modelos y diseños de robots
- Proporciona motivación a los estudiantes para programar la plataforma o diseñar y construir una nueva

En la Universidad Carlos III de Madrid se ha extendido la creación de robots imprimibles durante los últimos dos años. En la figura 1.2 se muestra el Miniskybot, un ejemplo de robot "open-source" desde la mecánica hasta la electrónica:

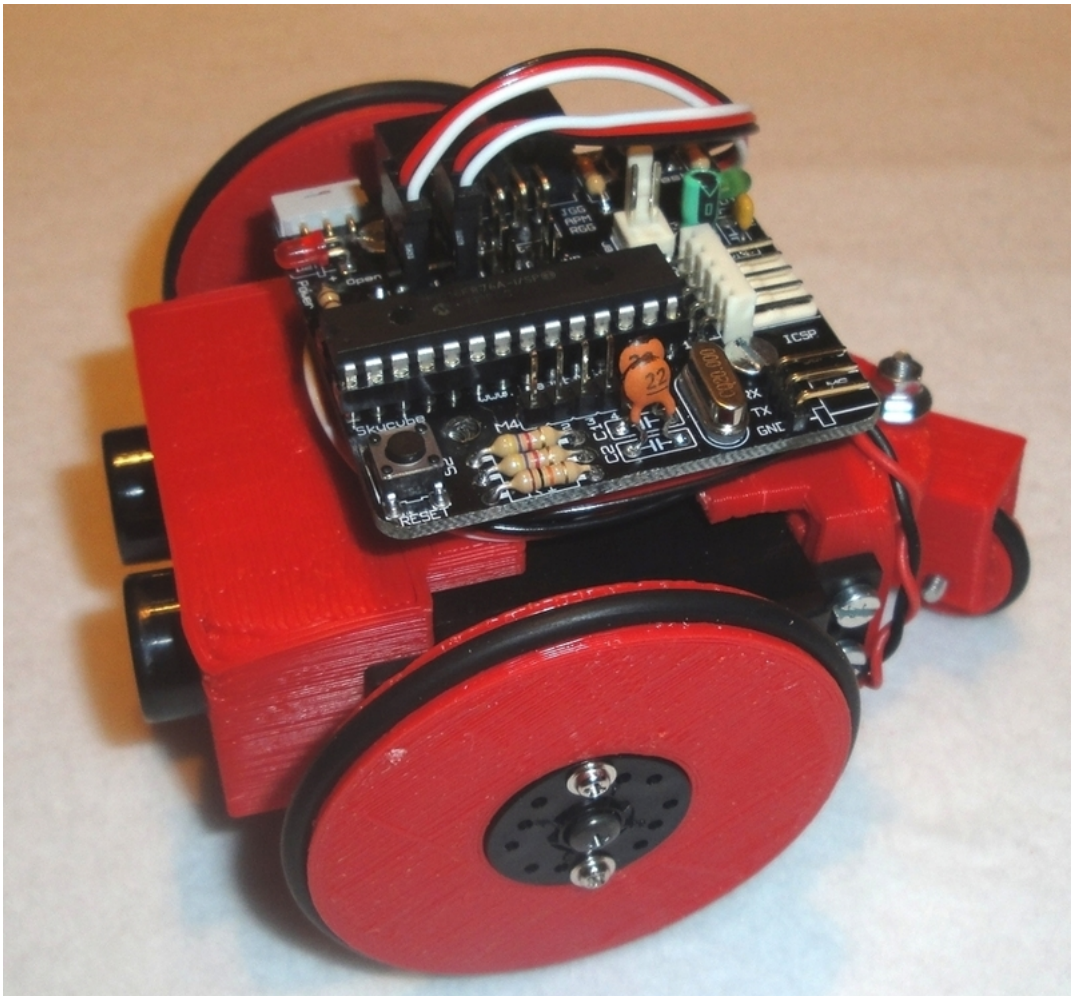

Figura 1.2: MiniSkyBot

1.2.2. Antecedentes

El predecesor directo de este proyecto es el F-Track. Este robot fue diseñado y construido en la UC3M por Jon Goitia, y posteriormente fue equipado y programado por Julián Marín.

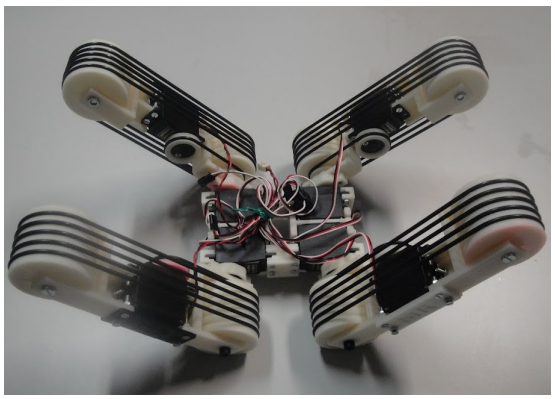

Figura 1.3: F-Track, el predecesor del D-Track

El F-Track presenta una morfología muy interesante: se mueve mediante orugas o "tracks", por lo que tiene las ventajas características de los vehículos con movimiento basado en ruedas o mecanismos similares; sin embargo, dado que los tracks están articulados, es posible sortear obstáculos que serían insalvables por un robot basado en ruedas. Este robot, no obstante, presentaba varios problemas:

- Todo el track está unido a la corona del servo, por lo que se produce un desequilibrio de ejes y se generan esfuerzos normales al plano de giro, lo que unido a la reducida potencia del servo genera una holgura de giro del track y una carencia de potencia en los servos de las articulaciones.
- La transmisión entre el servo y las ruedas se realiza mediante correas. En situaciones en las que se requiere un par motor elevado, la correa del motor puede patinar, lo que resulta en una pérdida de tracción.
- El sistema eléctrico y los servos de las articulaciones no proporcionan suficiente potencia, por lo que el levantamiento de los tracks es una operación lenta y aparatosa.
- Carece por completo de sistema de percepción, por lo que no puede implementarse un comportamiento autónomo del robot, dependiendo en todo momento del operador externo para evaluar la situación y operar el track.
- No proporciona suficiente capacidad de amplicación o modificación.

Por estos motivos se ha desarrollado el D-Track, objetivo del presente proyecto. El D-Track es un rediseño completo del F-Track que preserva la morfología y el movimiento basado en tracks, pero que trata de solucionar los problemas que han surgido y añadir características adicionales.

La figura 1.4 muestra el D-Track. Los problemas mencionados se han abordado de la siguiente forma en el D-Track:

- Se ha construido un eje fijo sobre el que rota el track. Se ha diseñado una transmisión mediante engranajes para la articulación del track.
- Se ha sustituido la transmisión entre el servo y las ruedas mediante correas por una transmisión basada en engranajes.

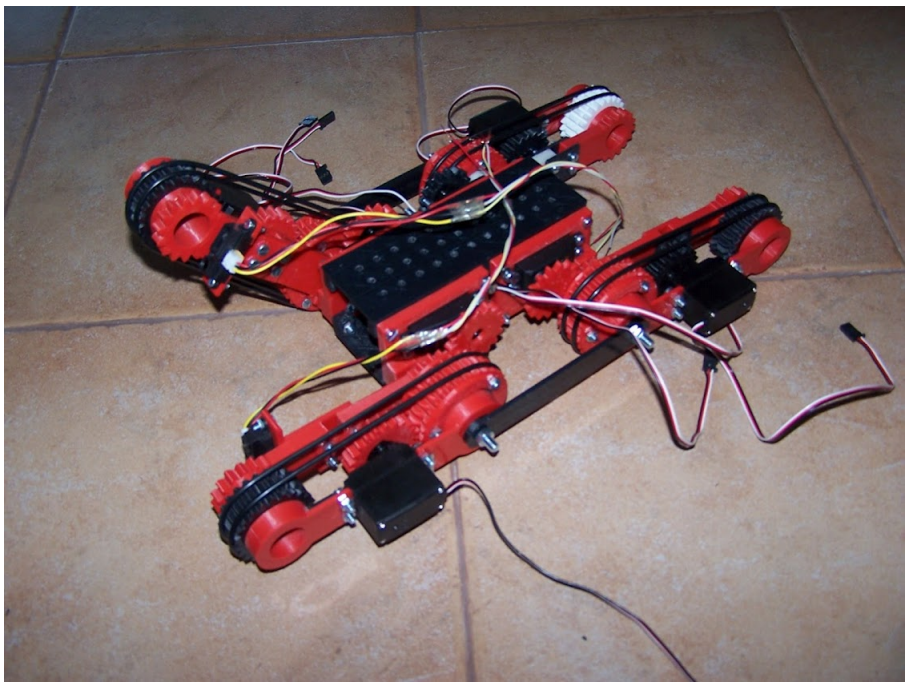

Figura 1.4: D-Track

- Se ha dotado el D-Track de una batería con amplia corriente de descarga y se han empleado servos más potentes en la articulación del track.
- Se ha dotado el robot de un sistema de percepción compuesto por sensores de distancia infrarrojos y un acelerómetro.
- Se ha diseñado teniendo en cuenta la capacidad de amplicación y modificación.
- Finalmente, se han hecho algunas optimizaciones al diseño. Por ejemplo, al emplear una rueda más pequeña en el extremo del track se reduce el momento de inercia en torno al eje del track.

1.2.3. Comparativa con alternativas no libres

En 2005, un grupo de investigación de la Universidad de Sherbrooke diseñó el robot AZIMUT [5]. Este robot presenta una morfología similar al F-Track y al D-Track, con un grado de libertad más en cada articulación de track. Una comparativa con el D-Track y el F-Track permite mostrar la gran capacidad de la robótica imprimible.

- Las dimensiones de AZIMUT son aproximadamente el doble que las del F-Track y el D-Track.
- El desarrollo del AZIMUT involucró a 14 personas: en el F-Track participaron 2 y en el D-Track una.
- El diseño mecánico de AZIMUT es cerrado y no replicable; y aunque fuera abierto, no sería fácilmente modificable. El diseño del D-Track es abierto, reproducible y modificable.
- El hardware y software de AZIMUT es también cerrado, mientras que el D-Track se basa en una plataforma popular de microcontroladores open-source (Arduino) y funciona con software libre.
- Desconocemos el coste de producción del prototipo de AZIMUT, pero podemos estimarlo en varios órdenes de magnitud más grande que el D-Track, que cuesta menos de 300 €

El AZIMUT ofrece más prestaciones en términos de potencia, velocidad, y funcionalidad: sin embargo, el F-Track y el D-Track ofrecen importantes ventajas derivadas, fundamentalmente, de su replicabilidad, y a un coste mucho menor. Este ejemplo ilustra la potencia del desarrollo de robots imprimibles open-source.

Figura 1.5: AZIMUT

1.3. Fases del proyecto

El proyecto se ha planificado en las siguientes fases, si bien en la práctica coexistían diferentes fases en determinados momentos.

1. **Análisis y brainstorming**: en esta fase se han estudiado los problemas del F-Track, se han propuesto soluciones, se han evaluado diversas alternativas de diseño y se han seleccionado las ideas de diseño más apropiadas.

2. **Formación**: en esta fase el autor se ha familiarizado con las tecnologías escogidas para elaborar el proyecto.

3. **Diseño y fabricación**: se trata de la fase en que se ha diseñado y fabricado la estructura física del robot. El motivo por el que el diseño y la fabricación forman parte de una misma fase es que el reducido tiempo de fabricación permite probar y evaluar los diseños de diferentes partes del robot en ciclos cortos, de modo que esta fase es un proceso iterativo compuesto por ciclos de diseño, fabricación y pruebas.

4. **Diseño del hardware**: a continuación, se han escogido los componentes eléctricos electrónicos a usar y se han diseñado las conexiones entre ellos.

5. **Estudio de la cinemática**: en esta fase se ha estudiado matemáticamente el robot y se han elaborado los modelos cinemáticos que describen la dinámica del robot en las situaciones de operación.

6. **Montaje**: durante este periodo se ha ensamblado el robot y se han conectado o soldado los componentes electrónicos.

7. **Programación**: aquí se ha elaborado el software de acuerdo a las leyes de control del vehículo, se ha calibrado los sensores, se ha determinado el protocolo de comunicación con el PC y se han desarrollado los algoritmos que implementan los comportamientos autónomos deseados.

8. **Pruebas**: en esta fase se ha probado la funcionalidad del robot y se han corregido los errores aparecidos.

9. **Redacción de la memoria**: se ha elaborado el texto y los diagramas de la memoria y se ha organizado el contenido de esta.

1.4. Tecnologías empleadas

1.4.1. Tecnologías libres

En el desarrollo de este proyecto se ha recurrido a tecnologías libres en la medida de lo posible. El motivo para ello es que uno de los objetivos del proyecto se basa precisamente en la filosofía open-source de los robots imprimibles: que sean fácilmente reproducibles y modificables. Por ello, una elección natural a la hora de seleccionar tecnologías con el objetivo de diseñar un robot open source es elegir tecnologías libres y accesibles, que no supongan una barrera a la hora de replicar el vehículo.

Breve historia del concepto de open source

Con la extensión del copyright al software en los años 70 y 80 y el crecimiento de la industria del software, la cantidad de restricciones aplicadas a la redistribución de software comenzó a crecer y empezó a generalizarse la distribución de binarios sin código fuente como medida para evitar la reproducción no autorizada de software. Frente a esta tendencia, nace en los 80 el concepto de *software libre* asociado a Richard M. Stallman, el proyecto GNU y la Free Software Foundation (FSF), creada en 1985. En 1986, la FSF publica la definición de software libre que mantiene hoy en día: el software es libre si el usuario tiene las siguientes libertades (teniendo en cuenta que las libertades 1 y 3 requieren el acceso al código fuente del software):

- Libertad 0: la libertad de emplear el programa para cualquier propósito
- Libertad 1: la libertad de estudiar el funcionamiento del programa y modificarlo para adaptarlo a las necesdades del usuario
- Libertad 2: la libertad de redistribuir el programa para ayudar al prójimo
- Libertad 3: la libertad de distribuir copias modicadas a terceros, de modo que dichas modificaciones puedan beneficiar a los demás

Hoy en día, una parte sustancial de la infraestructura de software que hace posible la sociedad de la información se basa en software libre. Los ejemplos más conocidos son el sistema operativo GNU/Linux, que se emplea en todo tipo de arquitecturas; Android (también basado en Linux), que domina el mercado de smartphones; el servidor web Apache, que opera en más de la mitad de los servidores web del mundo y los navegadores web Mozilla Firefox y Chromium[1], que ya son más usados que los navegadores privativos.

En 1998 nace la iniciativa OSI, que introduce el concepto de "open source" o "código abierto" buscando eliminar las ambigüedades en el término *free software* y popularizando el concepto en el mundo corporativo. Filosóficamente, el concepto de software libre se basa en la libertad del usuario, mientras que el concepto de "open source" enfatiza las ventajas prácticas del modelo. No obstante, en términos prácticas las definiciones de ambos términos son similares y la mayor parte de las licencias englobadas por uno de los conceptos son aceptadas por el otro y viceversa.

Hardware libre

El concepto de software libre no es fácil de transportar al hardware, especialmente si lo aplicamos al objeto físico. Si lo aplicamos a los planos del hardware, el concepto es menos problemático, siempre que tengamos en cuenta las restricciones de distribución que imponen algunos programas de CAD

[1] Google Chrome para Windows no es técnicamente software libre, pero gran parte del código es libre bajo el proyecto Chromium

1.4.2. Impresión 3D

Hoy en día, existen impresoras 3D comerciales y no comerciales con diversas características para diferentes ámbitos, empleando diversas tecnologías: sinterización selectiva por láser, modelado por deposición de fundente, estereolitografía o fabricación en láminas son algunas de las tecnologías actualmente existentes.

En este proyecto, vamos a emplear impresoras basadas en deposición de polímero fundente del tipo RepRap. El proyecto RepRap busca crear una plataforma de impresión open source accesible y de bajo coste, con el objetivo a largo plazo de lograr una máquina totalmente autorreplicable. Dado que estas impresoras están compuestas por piezas de plástico y pueden imprimir dichas piezas, una impresora RepRap es parcialmente autorreplicable.

Existen impresoras comerciales que, aunque rigurosamente no se pueden considerar RepRap debido a que las partes de la estructura no son generalmente imprimibles (por ejemplo, un marco de madera o de metal), están también ligadas al proyecto debido a que son compatibles a nivel de software y/o hardware. La impresora UC3-PO de la Asociación de Robótica de la Universidad en la que se han desarrollado los prototipos de las piezas, una Makerbot Thing-o-Matic, pertenece a este tipo de impresoras.

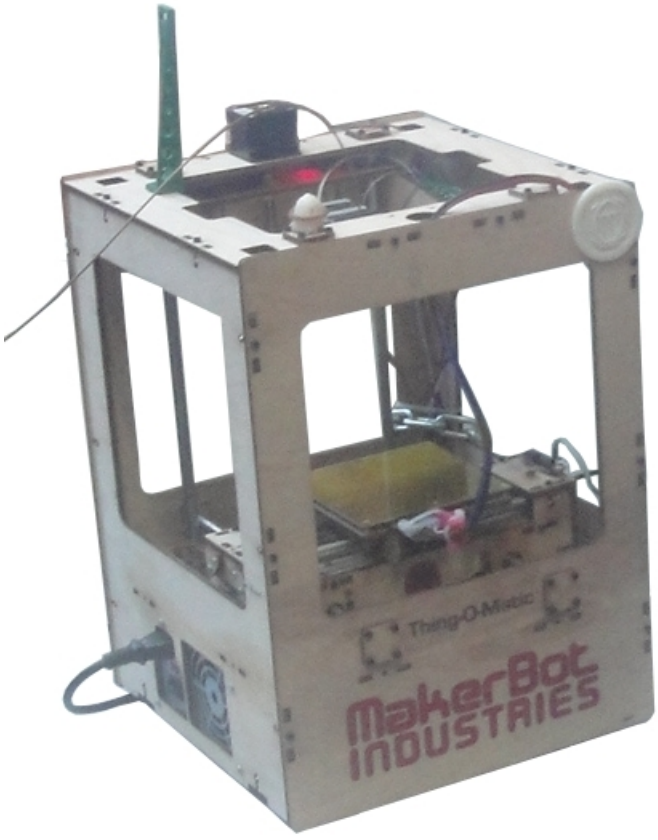

Figura 1.6: UC3-PO

La impresora en la que se ha imprimido gran parte del robot final, Mardan, es una Prusa Mendel (uno de los modelos de impresora RepRap más populares).

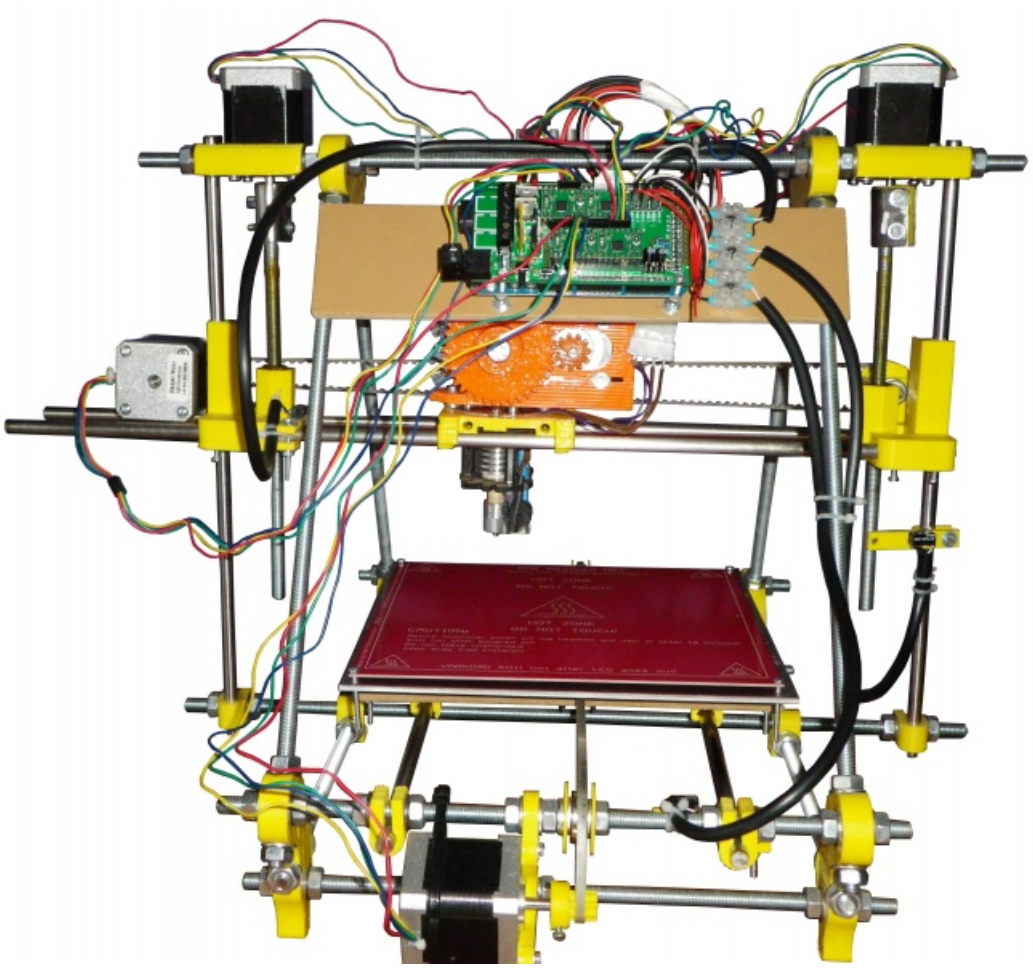

Figura 1.7: Mardan

Estas impresoras pueden emplear dos tipos de plástico: acrilonitrilo butadieno estireno (ABS) y ácido poliláctico (PLA). El presente proyecto se ha imprimido completamente en ABS, pero el PLA también tiene ventajas (como el hecho de requerir menores temperaturas de extrusión y ser biodegradable).

En estas impresoras, el extrusor deposita plástico fundente en sucesivas capas sobre una plataforma caliente para construir el objeto. En la mayoría de las configuraciones de impresora, el extrusor se mueve en uno o dos ejes cartesianos y la base se mueve en el resto.

1.4.3. Arduino

Arduino es una popular familia de microcontroladores open source basadas en microprocesadores Atmel AVR creada con el objetivo de proporcionar una plataforma electrónica accesible para todo tipo de proyectos. A día de hoy existen múltiples placas Arduino (así como placas compatibles desarrolladas por terceras partes) con diferentes microprocesadores, cantidad de pines de entrada/salida, tamaño y funcionalidad.

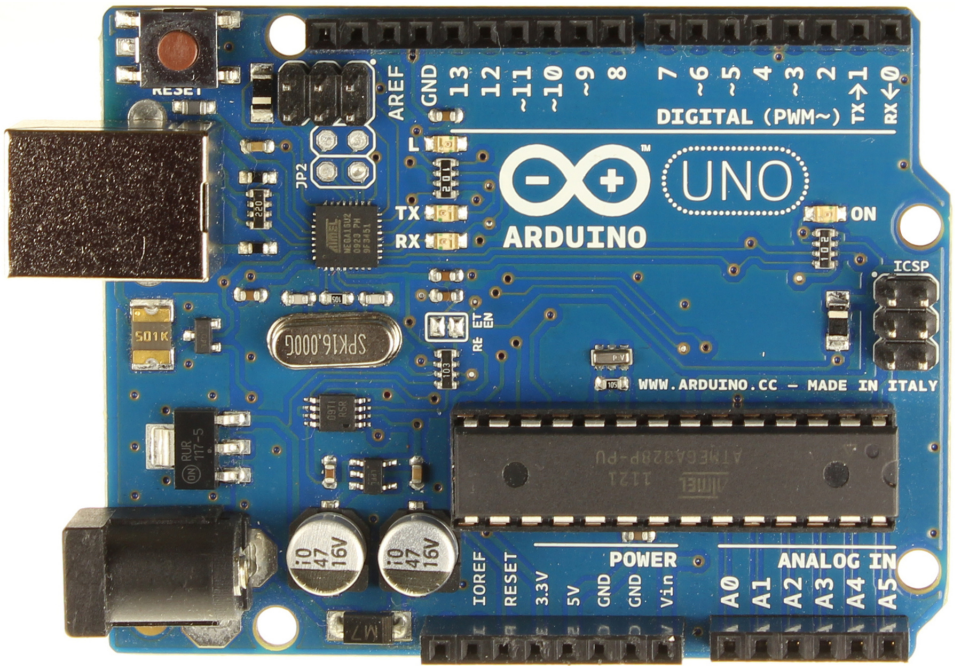

Figura 1.8: Arduino Uno

Arduino tiene una doble relación con este proyecto. Por una parte, la unidad de control del robot está compuesta por un Arduino Mega2560, un microcontrolador basado en el microprocesador ATMega2560. Por otra parte, la electrónica de las impresoras RepRap está también basada en la placa Arduino Mega2560 o placas derivadas.

Capítulo 2

Metodología de diseño

En cualquier proyecto de ingeniería, el diseño de los componentes físicos del objeto a crear es un proceso influenciado por una gran variedad de factores y sujeto a una serie de restricciones que dependen de consideraciones económicas, tecnológicas, ecológicas, logísticas o de cualquier otra índole. Por ello, a la hora de diseñar un proyecto es preciso realizar un análisis de estas circunstancias y determinar qué objetivos es preciso satisfacer en el proceso de diseño y a qué restricciones o consideraciones debe de estar sujeta esta actividad. Una planificación adecuada del diseño de acuerdo a este análisis permite prevenir posibles problemas en fases posteriores y optimizar el uso de tiempo y recursos humanos y no humanos en la elaboración del proyecto, así como satisfacer en mayor grado los objetivos del proyecto.

El presente capítulo se estructura de la siguiente forma:

- La sección 2.1 es un análisis de las circunstancias de diseño para el presente proyecto
- La sección 2.2 presenta unas pautas para el diseño orientado a la fabricación mediante tecnología de impresión 3D
- La sección 2.3 es un estudio de las operaciones posteriores a la fabricación y de su influencia sobre las decisiones de diseño
- La sección 2.4 es una lista de ejemplos de aplicación de esta metodología en el presente proyecto

2.1. Aspectos de diseño

El propósito de esta sección es explicar qué circunstancias se han tenido en cuenta a la hora de diseñar el presente proyecto. La tabla 2.1 enumera los diferentes factores que se han considerado, una descripción y una valoración cualitativa de su importancia relativa. La justificación de esta valoración y las consecuencias de diseño que tiene cada categoría se detallarán a continuación en la subsección correspondiente.

Concepto	Descripción	Relevancia	Impacto
Funcionalidad	Capacidad del robot de cumplir sus funciones mecánicas sin comprometer su integridad estructural, favoreciendo la operación de los motores	Alta	Alto
Fabricación	Proceso de elaboración de las diferentes piezas que componen el robot.	Alta	Alto
Producción	Organización y operación de las máquinas e instalaciones de producción y ensamblaje.	Baja	Nulo
Ensamblaje	Conjunto de operaciones que permiten combinar las piezas individuales en el robot final	Alta	Alto
Replicabilidad	Capacidad de una tercera parte para crear una réplica del robot	Alta	Bajo
Modificabilidad	Potencial de modificación de partes del diseño	Media	Medio
Mantenimiento	Posibilidad de reemplazo de piezas o partes individuales, o de calibrado	Media	Medio
Seguridad	Capacidad de evitar daños a seres vivos, otros objetos o al entorno	Baja	Bajo
Durabilidad	Capacidad de preservar la integridad del robot en circunstancias de operación, almacenamiento y transporte	Media	Bajo
Coste	Precio de la cantidad de material empleado en la construcción de las piezas	Baja	Bajo

Cuadro 2.1: Consideraciones de diseño

2.1.1. Funcionalidad

En todo proyecto de ingeniería, la medida en la que un diseño cumple los objetivos del proyecto es, naturalmente, la consideración de diseño más importante de todas. Es, por lo tanto, omnipresente en la fase de diseño. Un resumen de los ejemplos más significativos relativos a nuestro proyecto es la siguiente lista:

- Las extremidades deben presentar la movilidad deseada, girando en ángulos de entre -75º y +75º (en total, un arco de 150º)
- Las articulaciones deben presentar poco rozamiento y girar con facilidad
- Las uniones móviles deben presentar poca holgura
- Las piezas deben poder alojar los servomotores escogidos
- Las estructuras deben soportar los esfuerzos esperables en operación con una deformación inapreciable, incluyendo el peso de la electrónica de control.
- Las estructuras deben ser suficientemente ligeras y tener momentos de inercia suficientemente pequeños para ser operables mediante los servomotores.

2.1.2. Fabricación

Generalmente, las consideraciones de fabricación son las más importantes después de las de funcionalidad. Las características propias del proceso de impresión 3D deben ser tenidas en cuenta para obtener piezas válidas, lo que tiene un gran impacto sobre el diseño de las piezas del robot. Estas consideraciones requieren su propio apartado y, por ello, se detallan en la sección 2.2. En consecuencia, no se detallan aquí.

2.1.3. Producción

El objetivo del presente proyecto es crear un robot imprimible mediante una impresora 3D y ensamblado a mano de forma individual; es decir, no se busca crear un diseño que pueda ser producido de forma industrial, y por ello no se considera ningún tipo de instalación de producción y ensamblaje. Por lo tanto, su impacto sobre el diseño es nulo.

2.1.4. Ensamblaje

Al igual que la fabricación, el ensamblaje tiene un impacto significativo en el diseño. Por ello, es fundamental tener en cuenta qué operaciones de montaje y en qué orden se van a realizar a la hora de diseñar las piezas. En este proyecto se emplean los siguientes tipos de uniones:

- Unión mediante tornillos.
- Unión a presión.
- Unión mediante adhesivo basado en cianoacrilato.
- Unión mediante fusión con acetona.

Existe también cierta intersección entre el diseño para ensamblaje y el diseño para fabricación: por ejemplo, una pieza que es problemática para fabricar puede ser descompuesta en varias piezas ensambladas posteriormente.

2.1.5. Replicabilidad

El robot se ha diseñado de acuerdo a la filosofía de robots imprimibles de código abierto o de software y hardware libre. Uno de los principios de dicha filosofia es la libertad de réplica, de modo que cualquier usuario de un producto de software o hardware pueda crear una copia del mismo. Por ello, crear un robot fácilmente replicable es uno de los objetivos del proyecto y, en consecuencia, una consideración de diseño relevante.

El impacto sobre el diseño es, sin embargo, bastante reducido, ya que las tecnologías de diseño y construcción empleadas (OpenSCAD, impresoras 3D) son tecnologías que naturalmente favorecen este objetivo, por lo que no es necesario realizar decisiones de diseño adicionales para implementar la replicabilidad.

2.1.6. Modificabilidad

La capacidad para modificar el software o el hardware es otro de los principios de la idea de software y hardware libre y, adicionalmente, es un objetivo del proyecto. Al igual en el caso de la replicabilidad, las herramientas empleadas son adecuadas para crear diseños modificables y ampliables, por lo que la influencia sobre el diseño es moderada. Es, sin embargo, mayor que en el caso de la replicabilidad, ya que se ha buscando crear uniones y encajes genéricos que permitan emplear diferentes tipos de partes.

2.1.7. Mantenimiento

Se desea que el vehículo sea reparable y que las piezas dañadas sean reemplazables. Por ello, durante el diseño se preferirán las uniones desmontables (especialmente mediante tornillos y tuercas) y accesibles, de modo que sea posible desmontar parcialmente el robot para cambiar las piezas que sea necesario. Las consecuencias de diseño son significativas pero no son numerosas ya que el diseño para el ensamblaje y para la modificabilidad favorece naturalmente el mantenimiento.

2.1.8. Seguridad

La seguridad no cumple un papel muy relevante en el diseño por diversos motivos:

- El material de la estructura es plástico ABS, que no es tóxico. De hecho, se emplea en juguetes para niños (LEGO), ratones de ordenador y aparatos de higiene personal.
- La potencia de los servomotores es reducida, por lo que la capacidad de causar lesiones a personas o animales y la capacidad de causar desperfectos es prácticamente nula en caso de malfuncionamiento del robot.
- El robot no contiene ningún tipo de parte con riesgo de lanzamiento, explosión, movimiento brusco o fragmentación en esquirlas afiladas.

En consecuencia, el papel de las consideraciones de seguridad durante el diseño de este proyecto es pequeño.

2.1.9. Coste

La minimización de costes de material, que es fundamental en la industria y muchos campos, no cumple un papel destacado en este proyecto debido fundamentalmente a tres factores:

- El coste del ABS es reducido, en torno a 20 €/Kg
- Sólo se busca producir una unidad del robot
- Por motivos mecánicos, es deseable crear estructuras ligeras y, por lo tanto, con un contenido reducido de material. En consecuencia, un diseño que busque optimizar la funcionalidad puede favorecer la reducción de costes.

Por lo tanto, el coste no es una influencia importante en el diseño de las piezas.

2.2. Diseño para impresión 3D

2.2.1. Breve introducción a la tecnología

La creación de un objeto mediante una impresora 3D del proyecto RepRap consta de los siguientes pasos:

1. En primer lugar, se diseña el modelo de la pieza mediante un programa de CAD y se exporta a un archivo STL (`*.stl`). STL es el acrónimo de "Standard Tessellation Language" y es un formato de archivo muy extendido en aplicaciones relacionadas con la fabricación asistida por ordenador. La mayoría del software de CAD comercial y/o libre permite exportar modelos STL.

2. Posteriormente, se crea el `G-Code` a partir del STL mediante un software especializado (SkeinForge, Slic3r, RepSnapper). G-Code es un lenguaje de programación de máquinas CNC muy extendido. Un archivo de G-Code es una lista de instrucciones a ejecutar por la máquina. El software generalmente permite variar una gran cantidad de parámetros que determinan cómo se creará el modelo: por ejemplo, la fracción de relleno (una configuración muy empleada consiste en crear mallas, ahorrando material, creando piezas más ligeras y con una pérdida de propiedades mecánicas muy reducida), el grosor de la pared sólida, el redondeado de esquinas, el grosor de la capa, etcétera. Para este proyecto, los parámetros son, en la mayoría de los casos, muy próximos a los parametros por defecto.

3. Finalmente, la impresora crea la pieza ejecutando las instrucciones del G-Code. En esta fase interviene el firmware de la impresora y el software de comunicación con la impresora. En muchos casos, este software integra (ReplicatorG, Pronterface) o es el mismo software que el que lleva a cabo la fase anterior.

La figura 2.1 presenta un diagrama de este proceso.

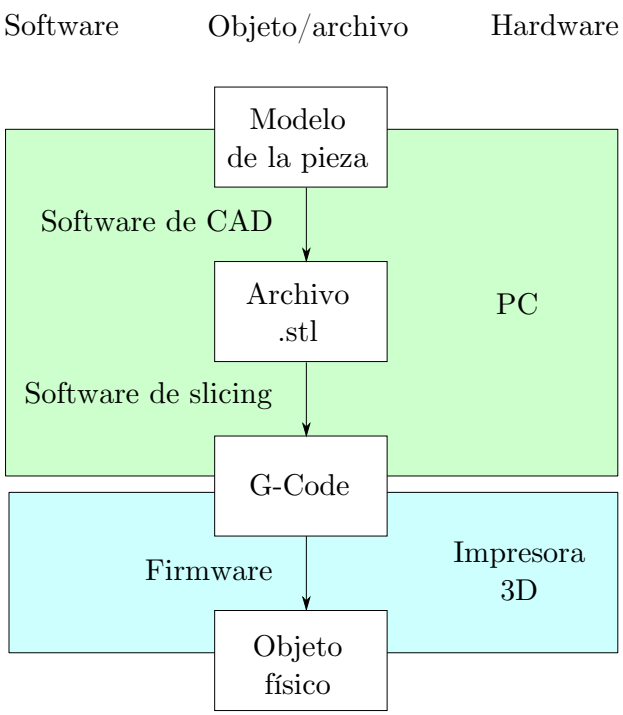

Figura 2.1: Toolchain de impresión 3D

Durante la creación de la pieza, el extrusor de la impresora está fundiendo un hilo del plástico, que va depositando en la plataforma caliente, formando capas apiladas. El posicionamiento de la punta del extrusor se realiza moviendo el extrusor o la plataforma caliente mediante motores paso a paso. Existen diferentes configuraciones, dependiendo de en qué ejes o planos se desplazan el extrusor y la plataforma. La mayoría de las impresoras RepRap populares son robots cartesianos: la Prusa Mendel, por ejemplo, desplaza en el eje Y la plataforma y en el plano XZ la herramienta, mientras que la Makerbot Cupcake y sus descendientes desplazan en el eje Z el extrusor y en el plano XY la plataforma. Existen configuraciones no cartesianas (por ejemplo, de tipo polar o de tipo SCARA), pero están menos extendidas.

2.2.2. Consideraciones de diseño

Rebabas de la base

En el proceso de impresión, la primera capa se adhiere directamente a la plataforma caliente (recubierta con cinta Kapton) y cada capa sucesiva se construye sobre la capa anterior. La consecuencia de este proceso es que la primera capa (en menor medida, las siguientes capas pueden presentar este fenómeno) queda sometida a condiciones de temperatura y presión significativamente diferentes a las de las demás capas. Estas condiciones favorecen el proceso de impresión al proporcionar adherencia adicional a la base (entendiendo "base" como la superficie de la pieza en contacto con la plataforma), lo que evita que se produzcan desplazamientos destructivos de material. Sin embargo, una consecuencia negativa de este fenómeno es que la primera capa tiende a ser, en mayor o menor medida, más *fina* en altura y más *ancha* en extensión, de modo que en una pieza de paredes rectas sobresale una rebaba de extensión inferior al milímetro en la altura de la base. Ello puede ser irrelevante en algunas piezas, pero no recomendable en otras, especialmente aquellas que tengan contacto mecánico con otras piezas (por ejemplo, engranajes).

Para abordar este problema, existen varias posibilidades:

- Diseñar la pieza de modo que la base no sea mecánicamente releveante: es decir, que la base no deba encajar, rodar o deslizar con otras piezas en el plano en el que sobresalen rebabas
- Configurar la impresión con *raft* a la hora de generar el G-Code. En la impresión con *raft*, la impresora imprime una base plana de plástico sobre la que imprime la pieza. Posteriormente, se debe separar a mano la pieza de la base de plástico. Dado que en este caso todas las capas quedan sometidas a las mismas condiciones (incluso la primera capa es imprimida sobre plástico ya sólido), la impresión con *raft* no crea rebabas
- Eliminar las rebabas posteriormente mediante un procedimiento de mecanizado como los detallados en la sección 2.3

En el presente proyecto, se ha empleado fundamentalmente la primera alternativa mencionada (imprimir las piezas de modo que la base no haga contacto con otras piezas); cuando esto no ha sido posible, se ha recurrido sobre todo a la tercera (eliminar las rebabas posteriormente)

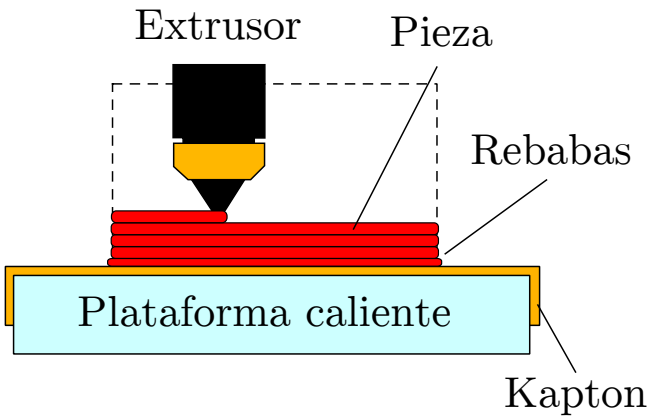

Figura 2.2: Aparición de rebabas

Sesgo del eje Z

Con cualquier tecnología de fabricación, las dimensiones de la pieza construida pueden diverger respecto a las del modelo. Si denominamos z a la dimensión teórica en el eje Z en el modelo, $\hat{z}$ a la dimensión de la pieza final, denominamos a la diferencia entre ambas $\epsilon(z) = z - \hat{z}$. Este error puede depender de la altura de diversas formas. Si aproximamos esta relación de dependencia linealmente:

$$\epsilon(z) = \epsilon_c + \epsilon_v z$$

Podemos denominar a ϵ_c *error fijo* o *error constante* y a ϵ_v *error variable*. En una impresora correctamente ajustada y calibrada, el error varianble ϵ_v debería ser inapreciable, por lo que para el diseño podemos asumir que $\epsilon_v \approx 0$. Sin embargo, debido a la distorsión de la primera capa explicada en el apartado correspondiente a las rebabas, puede aparecer un error fijo ϵ_c diferente de 0. Este fenómeno es especialmente relevante a la hora de diseñar encajes, uniones y holguras, por lo que se debe recurrir a una de las siguientes opciones de diseño:

- Diseñar la pieza de modo que la base no sea mecánicamente relevante. Al igual que en el caso de las rebabas, esta es una opción simple y efectiva.
- Estimar el sesgo (mediante la impresión de una pieza simple, calcular el error como $\epsilon_z = z - \hat{z}$) y compensarlo diseñando las piezas con mayor altura (si $z = z_0 + \hat{\epsilon}_z$, entonces $\hat{z} = z - \epsilon_z = z_0 + \epsilon_z - \epsilon_z = z_0$).
- Imprimir las piezas con una altura arbitrariamente superior y eliminar mediante un proceso de mecanizado el exceso de material hasta alcanzar la magnitud deseada

Se recomiendan fundamentalmente la primera y segunda alternativas, las más empleadas en la elaboración del robot. Para las impresoras empleadas, se ha estimado el sesgo del eje z en aproximadamente 1 milímetro.

Error en el plano XY

Al igual que en el caso del eje Z, las dimensiones X e Y se pueden desviar de los valores del modelo. Sin embargo, dichas desviaciones (suponiendo, de nuevo, una correcta calibración de la impresora) son de un valor significativamente inferior y causadas por la precisión limitada de la impresora, que depende fundamentalmente del diámetro de la boquilla del extrusor y de la precisión de los motores paso a paso. Al ser estas imprecisiones inferiores al milímetro no es posible medirla con medios convencionales y se recomienda, en cambio, un procedimiento basado en los siguientes pasos (siempre que sea necesario reducir esa imprecisión, lo cual sólo ocurre en unas pocas piezas):

1. Diseñar un modelo de las piezas a encajar en el que se elimina toda la pieza salvo una pequeña fracción de la sección o parte a encajar
2. Imprimir dicho modelo y verificar el encaje
3. En el caso de que el encaje no sea aceptable, variar las dimensiones del modelo de modo que creen un encaje con mayor o menor holgura y volver al paso 2. Si el encaje es correcto, continuar al paso 4
4. Ajustar las dimensiones del modelo de la pieza completa a las dimensiones del modelo de prueba

Debido a que el error en el plano XY no es fundamentalmente aleatorio sino que es un sesgo consistente, el procedimiento anterior genera resultados fiables.

Agujeros pasantes para varillas y tornillos

Por los mismos motivos que en el error en el plano XY, los agujeros en la pieza destinados a la inserción del tornillo u otra pieza cilíndrica no imprimible se desvían ligeramente de sus valores teóricos. Para diseñar los agujeros con las dimensiones correctas, se recomienda imprimir una pieza de baja altura (medio centímetro es suficiente) con una planta similar a la propuesta en la figura 2.3

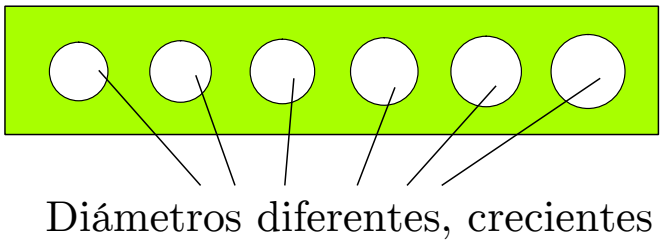

Figura 2.3: Pieza auxiliar para agujeros

Tras la impresión, se prueba a introducir el tornillo, varilla o pieza deseada en cada uno de los agujeros. Al diseñar las piezas finales, se dimensiona el agujero con el radio correspondiente al más pequeño de los agujeros en los que entra el objeto.

Se debe tener en cuenta que, al igual que en el exterior de la pieza pueden aparecer rebabas, en los agujeros paralelos al eje z también pueden aparecer rebabas en el interior del orificio, a la altura de la base. De nuevo, se recomienda mecanizar dichas rebabas.

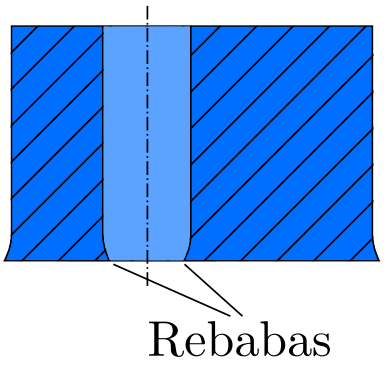

Figura 2.4: Rebabas interiores

Relaciones de dimensiones aceptables

En general, toda pieza en la que una de las dimensiones sea significativamente más grande que las otras dos puede resultar problemática en la práctica si es sometida a esfuerzos. Sin embargo, desde el punto de vista del diseño, una pieza de este tipo en la que la dimensión alargada sea la altura puede causar también problemas de impresión. La aparición de esfuerzos en la capa que se está extruyendo genera momentos perpendiculares al eje z que son proporcionales a la altura de la pieza. Si esos momentos son superiores al momento que genera la adherencia entre la superficie inferior de la pieza y la base caliente, se produce el despegue de la pieza de la plataforma y la pieza no se completará correctamente. El valor del ratio aceptable entre la altura y la longitud característica de la base depende de la impresora pero una buena regla, de acuerdo a las pruebas del autor, es no imprimir piezas con ratios superiores a 8:1. Si es posible, una solución trivial es rotar la pieza e imprimirla con una de las caras largas como base.

Impresión en pendiente

En impresoras de extrusor único basadas en la deposición de plástico fundente puede resultar problemática la impresión en pendiente en piezas en las que la sección aumenta con la altura, ya que cada capa debe apoyarse en la anterior. Las pendientes con poca inclinación no representan problema para impresión, pero a partir de cierto valor puede comenzar a deformarse la pieza. En general, una regla comúnmente aceptada es que las pendientes de 45º e inferiores son seguras para imprimir. Las recomendaciones de diseño para evitar problemas de pendiente son:

- Diseñar piezas prismáticas con sección constante
- Imprimir las piezas con la sección ancha como base
- Dividir la pieza en varias partes y unirlas mediante un procedimiento de la sección 2.3.
- "Rellenar" parte o todo el hueco en el diseño y eliminar el material sobrante posteriormente. De nuevo, véase la sección 2.3

Si, a pesar de ello no es posible eliminar una pendiente negativa, es posible que, aún así, la impresión sea moderadamente satisfactoria siempre que la parte con pendiente o arco sea suficientemente pequeña. Por ello, a pesar de que se recomienda que los agujeros pasantes se impriman paralelos al eje Z, los agujeros en pared pueden ser viables si no son excesivamente grandes (si bien no alcanzan la misma calidad que los agujeros con la orientación recomendada)

2.3. Operaciones posteriores a la fabricación

El propósito de esta sección es explicar las operaciones de mecanizado y unión recomendadas para las piezas creadas en ABS mediante una impresora RepRap. Aunque las operaciones de unión son generalmente relevantes en el diseño para el montaje, en ocasiones están relacionadas con el diseño para fabricación al crear piezas.

2.3.1. Corte con cúter

El ABS es un plástico relativamente blando, comparado con otros materiales de construcción: por ello, es fácilmente mecanizable con diversas herramientas. Una herramienta útil para trabajar con piezas de ABS es el cúter. El cúter permite recortar partes no deseadas con grosor reducido, como las rebabas de la base (como se menciona en la sección 2.2). Otra operación común con cúter es la eliminación del *raft*.

2.3.2. Lijado y esmerilado

El lijado es una operación efectiva en piezas de ABS para los siguientes propósitos

- Eliminación de rebabas e irregularidades.
- Ajuste de dimensiones mediante la eliminación de material
- Alisado de superficies
- Ampliación de agujeros (herramienta rotativa)

Dado que es un material blando, se recomienda emplear lija de grano fino. Si se desea eliminar una cantidad significativa de material es posible emplear grano medio o grande (posiblemente con una herramienta rotativa de tipo "Dremel"), finalizando el proceso con una lija fina.

2.3.3. "Barnizado" con acetona

El plástico ABS se puede disolver con acetona. Por ello, la acetona es útil para diversos propósitos; uno de ellos consiste en la aplicación de acetona a la superficie de la pieza mediante un pincel, como si se estuviera pintando la pieza. Esta operación causa una fusión superficial entre las capas, aumentando significativamente la dureza de la superficie y, moderadamente, la tenacidad de la pared de la pieza. Adicionalmente, tras aplicar acetona, las iregularidades en la superficie de la pieza se reducen, dando como resultado una superficie más lisa.

2.3.4. Pegado con acetona

Otra aplicación efectiva de la acetona es la unión de piezas. Para asherir piezas con acetona, se debe aplicar una cantidad reducida de acetona a modo de pegamento sobre las superficies a unir, por ejemplo mediante un pincel. A continuación, se deben unir las piezas antes de que la acetona se seque y se debe dejar fundir unos segundos. En menos de un minuto, el proceso estara completo. La unión con acetona es bastante fuerte, incluso con poca superficie de contacto. Por su reducido precio por litro en comparación con los pegamentos especializados, además de su gran efectividad, se recomienda siempre emplear acetona para pegar piezas de ABS con otras piezas de ABS.

2.3.5. Pegado con pegamentos genéricos para plásticos

Los pegamentos basados en cianoacrilato también son efectivos a la hora de pegar piezas de ABS entre sí y con otros plásticos. Como se ha explicado en la sección 2.3.4, los pegamentos comerciales son una alternativa inferior a la acetona a la hora de unir piezas de ABS entre sí. Sin embargo, la acetona puede no ser efectiva a la hora de unir piezas de ABS con otros plásticos o materiales; en este caso, se recomienda recurrir a dichos adhesivos. Las propiedades mecánicas de la unión dependeran del pegamento usado y del material adherido.

La principal desventaja para este proyecto tanto de la unión con acetona

2.3.6. Unión mediante tornillos

Los tornillos son una solución universal para uniones en todo tipo de aplicaciones, y el montaje de robots basados en piezas de ABS es una aplicación más. Los tornillos presentan las ventajas de las uniones permanentes (una unión con tornillo y tuerca puede durar indefinidamente) y las de las uniones no permanentes (es posible desmontar la construcción, lo que permite reparar o modificar el robot con facilidad); por ello, en este proyecto se han escogido uniones con tornillos en la medida de lo posible. El diseño orientado a la unión con tornillos es sencillo, y la reducida dureza del plástico ABS permite que una unión con tornillo y tuerca apretada adquiera una fuerza considerable sin el uso de arandelas, ya que, aplicando una fuerza moderada, la cabeza del tornillo y la tuerca pueden deformar ligeramente la superficie de la pieza y "hundirse" en ella, lo que incrementa la fuerza de la unión.

2.3.7. Unión a presión

Las piezas de ABS se pueden unir a presión, siempre que la intereferencia (la diferencia entre diámetros) tenga una magnitud suficiente. Diseñar una unión a presión es complicado y requiere un proceso de prueba y error, similar al descrito en la sección 2.2.2. Dependiendo de la holgura final, la unión a presión puede ser prácticamente permanente o desmontable (si bien una unión desmontable puede no ofrecer la fuerza que se requiere). Para uniones permanentes, es preferible realizar una unión con acetona en lugar de una unión a presión, mientras que para una unión desmontable es recomendable emplear tornillos. No obstante, en situaciones específicas, una unión a presión puede ser una solución adecuada.

2.4. Ejemplos

2.4.1. Diseño de la articulación y de la pata

Con la finalidad de posibilitar el montaje del rodamiento y facilitar la impresión, las partes de la pata y su articulación se han dividido en dos partes: una parte interior y otra parte exterior. En la figura 2.5 se puede observar un diagrama de las dos piezas que conforman una mitad de la extremidad. Ambas partes quedan finalmente unidas mediante acetona, de la forma explicada en la sección 2.3.4

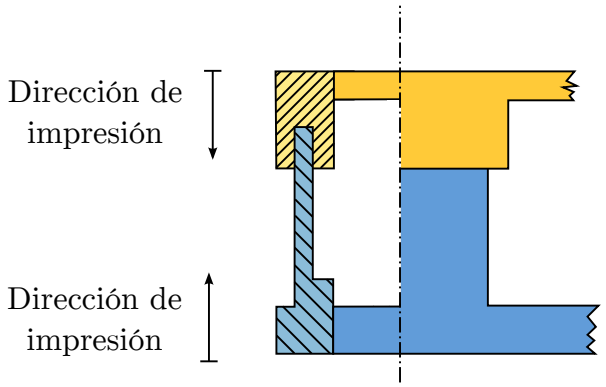

Figura 2.5: Parte de la extremidad

2.4.2. Unión de los engranajes con la corona

Para la unión de las coronas recortadas de los servos con los engranajes, tanto los de la rueda como los de la pata, se ha combinado una unión a presión con una unión mediante pegamento para plásticos genérico, ya que la corona del servo es de otro tipo de plástico. De este modo, los esfuerzos no son sólo soportados por una unión (ya sea la unión a presión o la unión con adhesivo), sino que se reparten entre las dos de modo que cada unión soporta los esfuerzos más favorables.

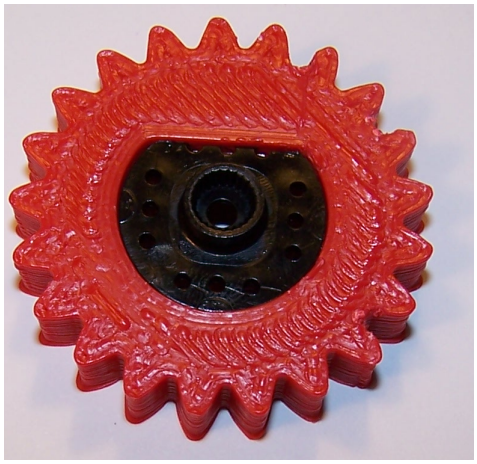

Figura 2.6: Unión de engranaje y corona

2.4.3. Uniones con tornillos

Como se justifica en la sección 2.3.6, las uniones mediante tornillos conforman la mayoría de las uniones de piezas en este proyecto. Todos los tornillos empleados son tornillos de rosca métrica M4, exceptuando los tornillos empleados para fijar los sensores de infrarrojos (M3). Para diseñar los agujeros del tamaño adecuado, se empleó el procedimiento explicado en la sección 2.2.2, apartado "Agujeros pasantes para varillas y tornillos". Para la Makerbot Thing-o-Matic de la Asociación de Robótica de la UC3M en la que se realizaron las piezas prototipo, un agujero de 2.25 mm de radio es suficiente para crear un agujero para tornillo M4, mientras que para la Prusa Mendel en la que se imprimió la mayoría del robot final se realizó un agujero de 2.5 mm de radio.

2.4.4. Diseño de los rodamientos

Para crear una articulación basada en rodamientos con la mínima holgura posible es imprescindible tener en cuenta el sesgo del eje z, como se describe en la sección 2.2.2, apartado "Sesgo del eje Z". Este sesgo se compensa en el diseño de las piezas, de modo que el resultado final es la eliminación de ese sesgo y la obtención de una articulación móvil con una holgura muy reducida.

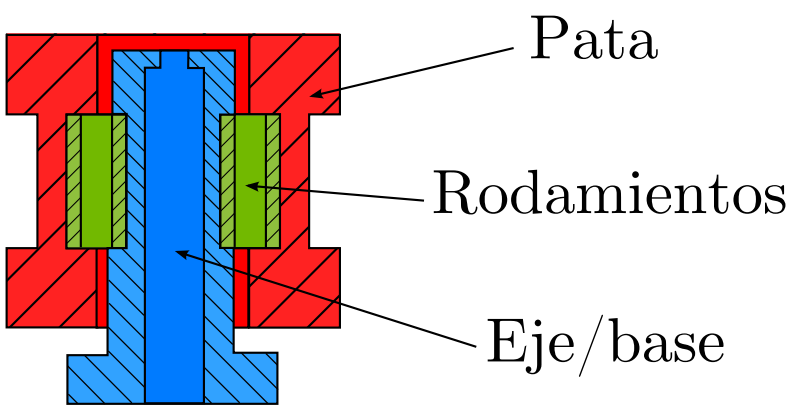

Figura 2.7: Unión de engranaje y corona

2.4.5. Diseño de las ruedas

Las ruedas grandes son piezas cilíndricas (con forma de engranaje en la superficie de rodadura), huecas por el interior y con diámetros interiores variables: el centro de la rueda tiene un diámetro interior superior al de las tapas que cierran la rueda de un lado y de otro. Por lo tanto, no es posible imprimir la rueda grande de una pieza y se ha separado en tres partes, una por cada sección diferente: dos tapas o llantas idénticas y una pieza central, que comprende la mayor parte la rueda. De este modo, estas partes son imprimibles y la rueda se puede montar a partir de ellas. La unión entre las piezas se realiza con tornillos M4 de 30 mm de largo, de modo que atraviesan la totalidad del conjunto.

Capítulo 3

Descripción del sistema mecánico

En todo tipo de robots, desde los robots industriales hasta los vehículos autónomos, su configuración mecánica determina las capacidades del robot. Tanto la estructura como las articulaciones, los elementos de transmisión y los elementos motrices deben estar diseñados de forma que el robot pueda llevar a cabo sus funciones de forma efectiva y eficiente.

Este capítulo está compuesto por las siguientes secciones:

- La sección 3.1 describe la morfología del robot y los elementos mecánicos que la componen.
- La sección 3.2 presenta el modelo geométrico y cinemático del robot.
- La sección 3.3 presenta el control de velocidad de las ruedas en navegación y maniobra.

3.1. Descripción de la mecánica del robot

La figura 3.1 presenta las principales partes de la estructura del robot y señala la posición de algunos elementos significativos:

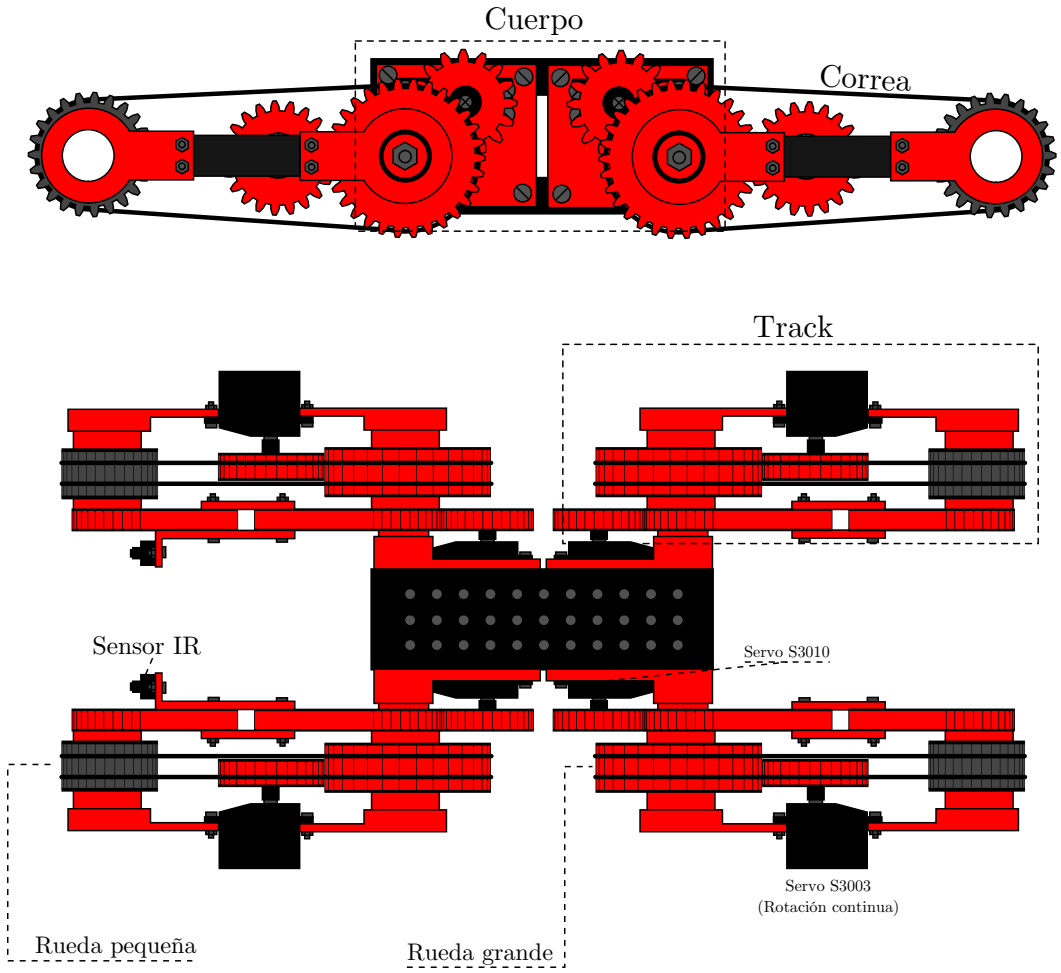

Figura 3.1: Ilustración esquemática del robot

El robot está compuesto por un cuerpo central y 4 tracks, unidos mediante una articulación basada en rodamientos cilíndricos lineales. Estos tres elementos, cuerpo, pata y articulación, se describen en las subsecciones 3.1.1, 3.1.2 y 3.1.3 respectivamente.

3.1.1. Descripción del cuerpo

El cuerpo está formado por dos bases (pieza negra) y cuatro piezas del eje. Las bases son idénticas, de tal forma que una se monta en la parte superior y otra en la parte inferior de forma simétrica, e incluyen una matriz de agujeros para tornillos M4, destinados a la instalación de elementos adicionales o a la ampliación del robot. Las piezas del eje se colocan en los laterales, e incluyen un hueco para ubicar los servos Futaba S3010, cuya misión es levantar los tracks, y un eje sobre el que rota el track, que es hueco para permitir el paso del tornillo M6 que cierra el eje con el tapón.

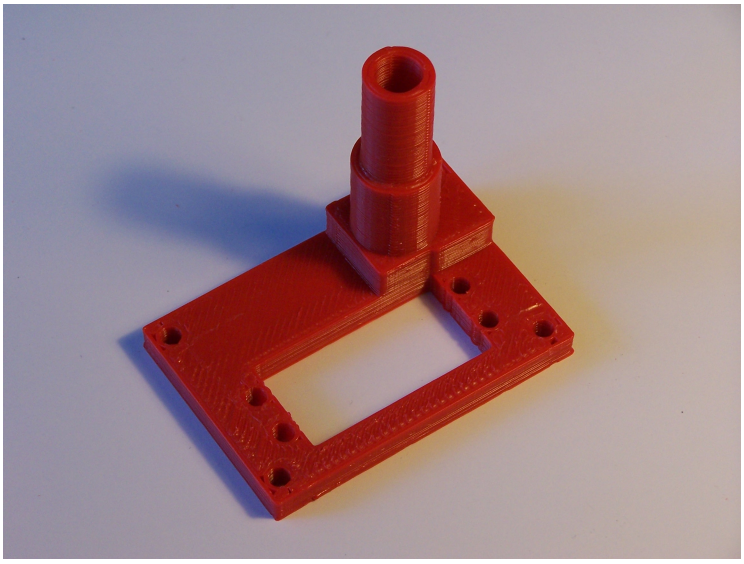

Figura 3.2: Pieza del eje

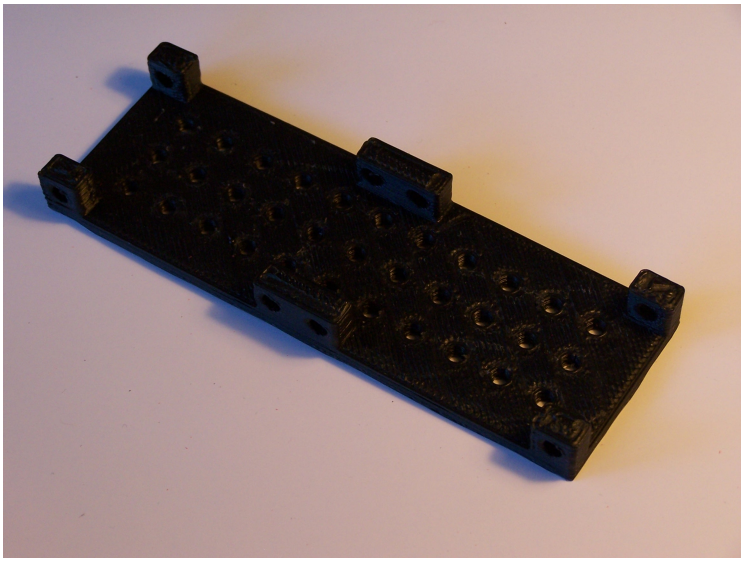

Figura 3.3: Pieza de la base

3.1.2. Descripción del track

La estructura del track está formada por las piezas exteriores, interiores y un Futaba S3003 modificado para rotación continua, cuya función es hacer girar las ruedas. Se ha diseñado de forma que este servo es parte de la estructura. Sobre el track van montadas las ruedas: la rueda pequeña gira directamente sobre la parte cilíndrica de las piezas interiores, mientras que la rueda grande gira sobre un asiento, también cilíndrico. Ambas ruedas van unidas mediante una correa de goma de 377 mm de perímetro y 3 mm de diámetro.

Las piezas interiores son parcialmente engranajes, lo que permite que el engranaje del track, unido a la corona del servo montado en el cuerpo, engrane con el track y su giro lo levante o haga descender. Las ruedas también son engranajes, y la rueda grande queda engranada con el engranaje de la rueda, unido al servo que va montado en el track. Todos los engranajes del robot son engranajes con un diente de perfil de envolvente del círculo, con un ángulo de presión de 28°

La relación de transmisión en el par de engranajes que mueven el track es 1:1, ya que ambos engranajes tienen el mismo diámetro y las mismas dimensiones de dientes. En el par de engranajes de la rueda, la relación de transmisión es 21:30, ya que el engranaje motriz tiene 21 dientes y el engranaje conducido (la rueda) tiene 30.

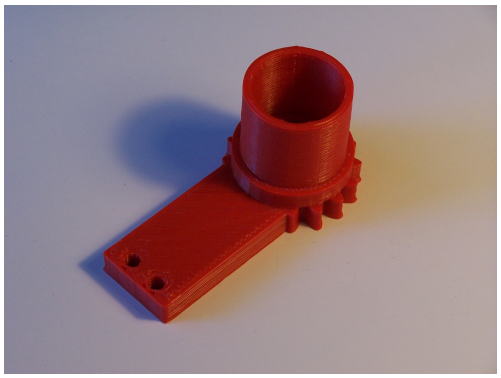

Figura 3.4: Pieza interior del track (larga)

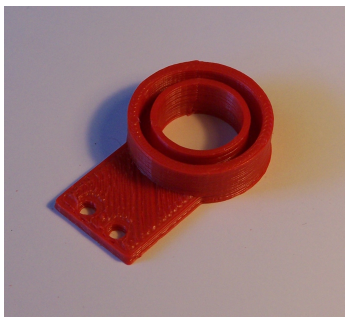

Figura 3.5: Pieza exterior del track

3.1.3. Descripción de la articulación

La articulación que une el track con el cuerpo está basada en rodamientos cilíndricos lineales huecos, ya que es una unión que tiene que soportar mayores esfuerzos que las demás y, por lo tanto, aumenta la relevancia de la reducción del rozamiento. En la figura 3.6 se puede observar un esquema de la articulación.

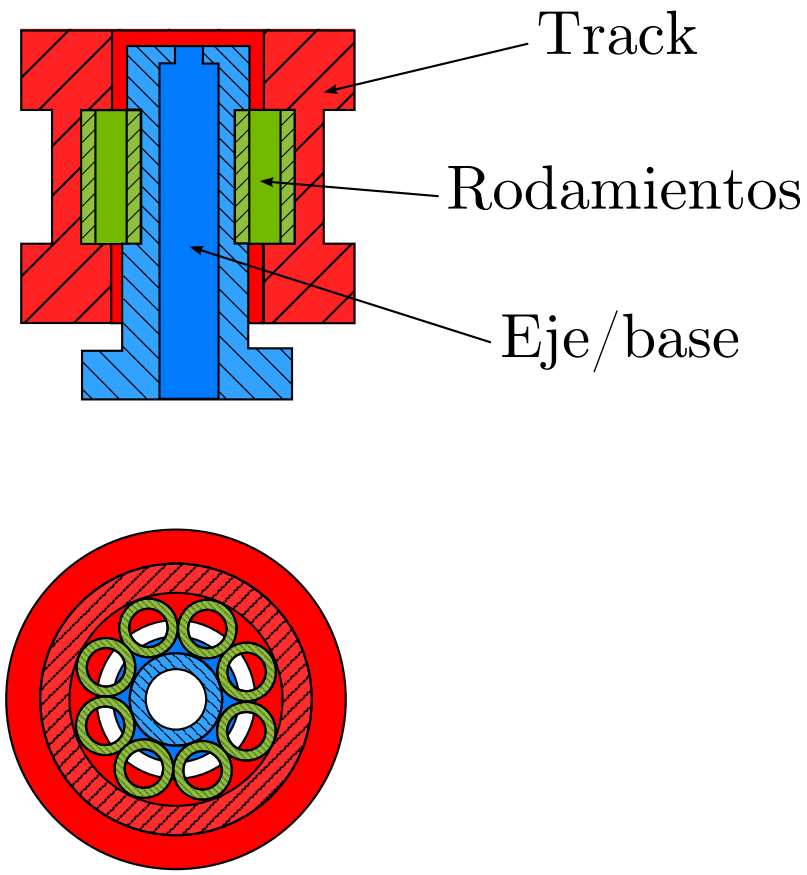

Figura 3.6: Rodamientos

3.2. Modelo cinemático del robot

La figura 3.7 presenta un esquema geométrico del robot en el sistema de referencia del robot.

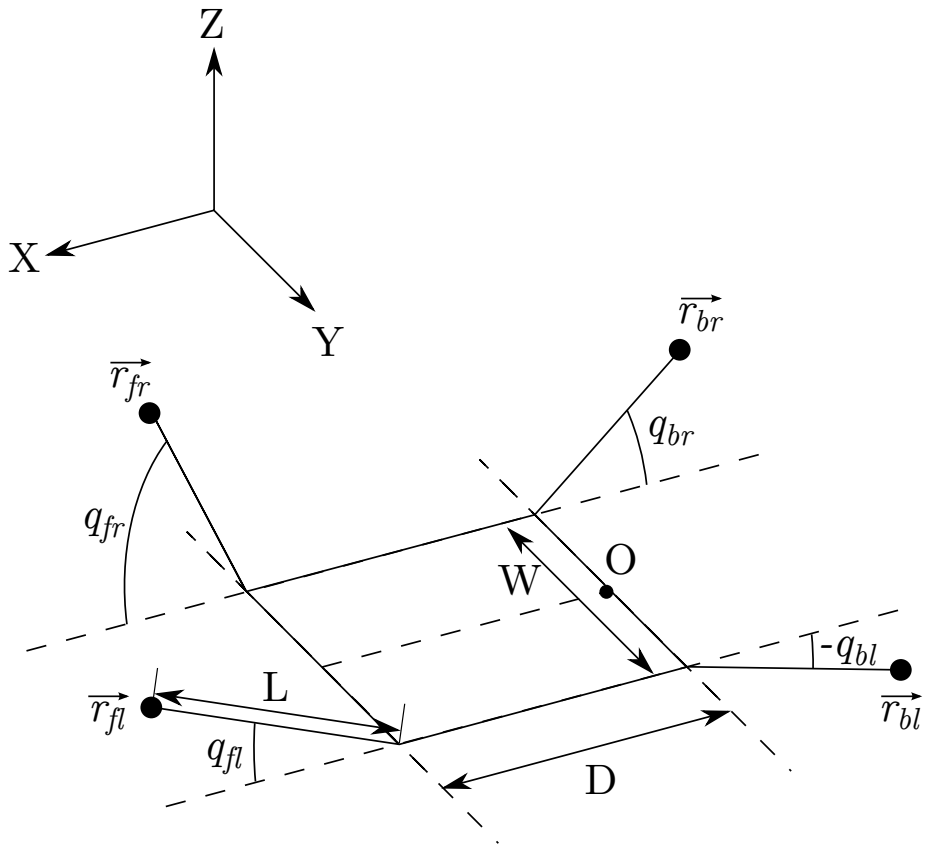

Figura 3.7: Diagrama de la morfología del robot

Las distancias son D = 11.5 cm, L = 12 cm y W = 14 cm. En este modelo, la posición de los extremos en el sistema de referencia del robot viene dada por las ecuaciones 3.1 a 3.4

$$\vec{r}_{fr} = \vec{r}_O + \begin{bmatrix} \mathrm{D} \\ -\mathrm{W}/2 \\ 0 \end{bmatrix} + L \begin{bmatrix} \cos q_{fr} \\ 0 \\ \sin q_{fr} \end{bmatrix} \tag{3.1}$$

$$\vec{r}_{fl} = \vec{r}_O + \begin{bmatrix} \mathrm{D} \\ \mathrm{W}/2 \\ 0 \end{bmatrix} + L \begin{bmatrix} \cos q_{fl} \\ 0 \\ \sin q_{fl} \end{bmatrix} \tag{3.2}$$

$$\vec{r}_{br} = \vec{r}_O + \begin{bmatrix} 0 \\ -\mathrm{W}/2 \\ 0 \end{bmatrix} + L \begin{bmatrix} -\cos q_{br} \\ 0 \\ \sin q_{br} \end{bmatrix} \tag{3.3}$$

$$\vec{r}_{bl} = \vec{r}_O + \begin{bmatrix} 0 \\ \mathrm{W}/2 \\ 0 \end{bmatrix} + L \begin{bmatrix} -\cos q_{bl} \\ 0 \\ \sin q_{bl} \end{bmatrix} \tag{3.4}$$

Proporcionaremos un quinto grado de libertad al robot posicionándolo en el eje X mediante las ruedas, según la ecuación 3.5. La operación de los motores de las ruedas para obtener q_x se cubre en la sección 3.1.3.

$$\vec{r}_O = \begin{bmatrix} q_x \\ 0 \\ 0 \end{bmatrix} \tag{3.5}$$

Para obtener la posición del extremo del robot respecto al sistema de referencia de la superficie, debemos multiplicar la posición del extremo del robot en el sistema de referencia del robot por las inversas de las matrices de rotación que determinan la orientación del sistema de referencia del robot, de acuerdo a la ecuación 3.6. Estas matrices vienen descritas en la ecuación 3.7.

$$\vec{r}_{superficie} = R_x^{-1}(\phi) R_y^{-1}(\theta) R_z^{-1}(\psi) \vec{r}_{robot} \tag{3.6}$$

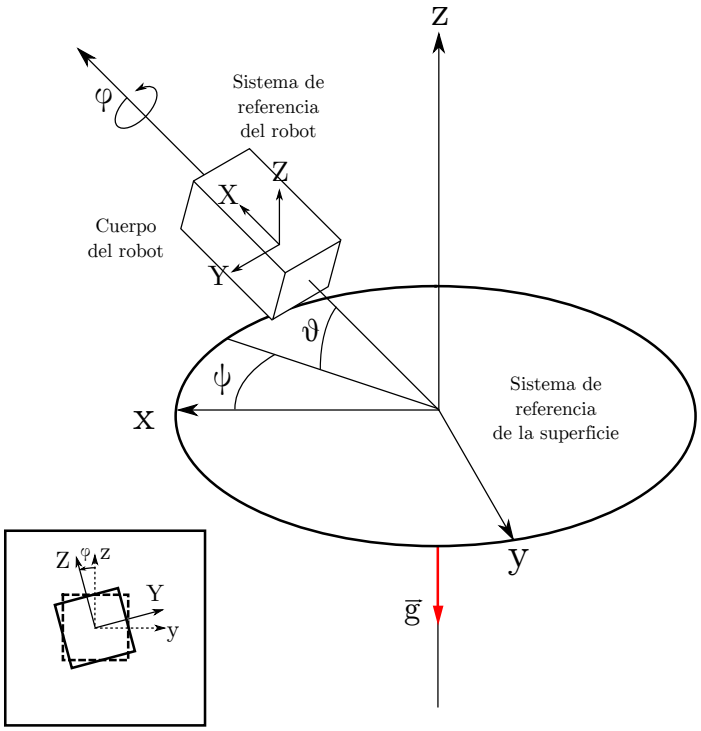

Figura 3.8: Sistemas de referencia

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} ; R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} ; (\theta)R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

Dado que la orientación del robot es una variable endógena al estado del robot (q_{fr}, q_{fl}, q_{br}, q_{bl}, q_x) y al relieve de la superficie, es teóricamente posible obtener el modelo cinemático completo del robot que nos permite obtener la posición de cualquier punto del robot respecto a un punto fijo en superficie. Sin embargo, sería una tarea de elevada complejidad, dependiente de una caracterización analítica del relieve $h(x, y)$ y de reducida relevancia práctica para este proyecto, ya que el sistema de percepción del robot tiene capacidad limitada y obtener tal

caracterización resulta prácticamente imposible. En cambio, un enfoque más productivo consiste en obtener modelos cinemáticos simplificados para las situaciones de interés, aplicando una serie de restricciones dependientes de la situación. La sección 3.2.1 presenta un ejemplo de dicho modelo.

3.2.1. Modelo cinemático simplificado

Aplicando las siguientes restricciones, que están presentes en una cantidad relevante de casos de operación del robot, podemos simplificar el modelo del robot:

- Los tracks frontales tienen el mismo ángulo: $q_{fr} = q_{fl} = q_f$
- Los tracks traseros tienen el mismo ángulo: $q_{br} = q_{bl} = q_b$
- El relieve sólo varía en el eje x: $\frac{\partial}{\partial y} h(x, y) = 0 \ \forall (x, y) \in \mathbb{R}^2$
- El ángulo de guiñada ψ es 0

Como consecuencia de estas restricciones, el ángulo de alabeo ϕ también es 0. La figura 3.9 ilustra el modelo simplificado:

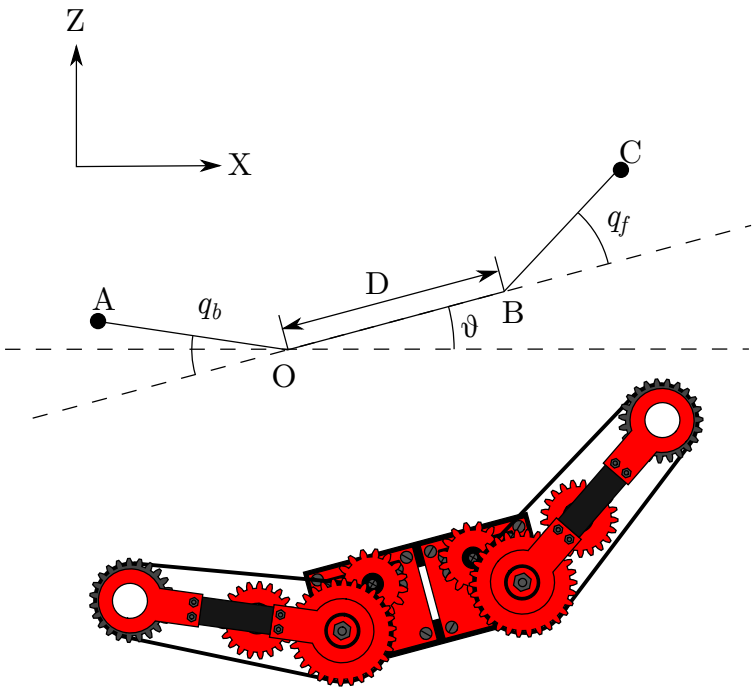

Figura 3.9: Modelo morfológico simplificado

El modelo viene descrito por las ecuaciones 3.8 a 3.11.

$$\vec{r}_A = \vec{r} + \begin{bmatrix} -\cos(q_b - \theta) \\ \sin(q_b - \theta) \end{bmatrix} \mathrm{L} \tag{3.8}$$

$$\vec{r}_B = \vec{r}_O + \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \mathrm{D} \tag{3.9}$$

$$\vec{r}_C = \vec{r}_B + \begin{bmatrix} \cos(\theta + q_f) \\ \sin(\theta + q_f) \end{bmatrix} \mathrm{L} \tag{3.10}$$

$$\vec{r}_O = q_x \tag{3.11}$$

Aplicando la restricción correspondiente al terreno es posible determinar el valor de θ en función de q_x, q_f y q_b, con lo que obtendremos el modelo simplificado correspondiente a la situación de operación en que nos encontramos. Finalmente, para aplicaciones prácticas, el uso de restricciones adicionales permite reducir el número de grados de libertad a 2. A continuación se presentan algunos ejemplos:

Posicionamiento del extremo frontal en superficies planas

En este caso de operación, operaremos sobre una superficie plana y fijaremos $q_b = 0$ como restricción adicional para reducir grados de libertad. Por lo tanto, el ángulo de cabeceo es nulo ($\theta = 0$) y la posición del extremo viene dada por la ecuación:

$$\vec{r}_C = \begin{bmatrix} x_c \\ z_c \end{bmatrix} = \begin{bmatrix} q_x + D + \mathrm{L}\cos q_f \\ \mathrm{L}\sin q_f \end{bmatrix}$$

El espacio de trabajo es $\{(x_c, z_c) \in \mathbb{R}^2 |\ 0 \leq z_c \leq \mathrm{L}\sin 75°\}$, ya que los ángulos de track están limitados a 75° por diseño.

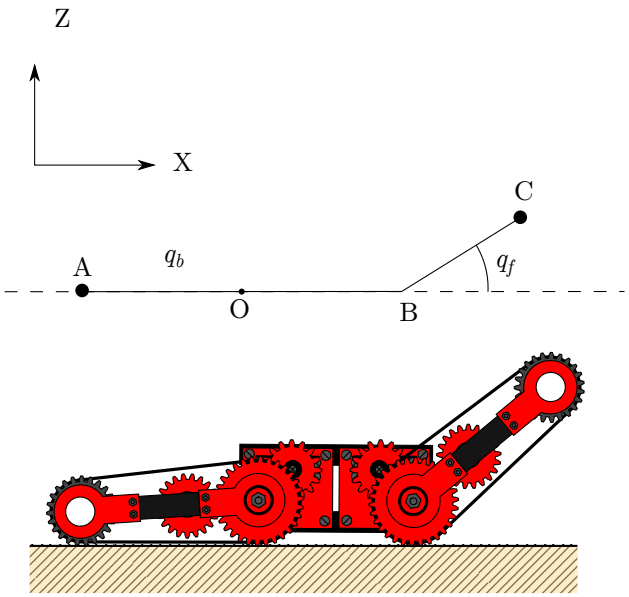

Figura 3.10: Posicionamiento del punto C

Por lo tanto, la cinemática inversa del robot viene dada por las ecuaciones:

$$q_f = \arcsin \frac{z_c}{\mathrm{L}}$$

$$q_x = x_c - D - \sqrt{\mathrm{L}^2 - z_c^2}$$

El Jacobiano que verifica $\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \end{bmatrix} = J \begin{bmatrix} \dot{q}_x \\ \dot{q}_f \end{bmatrix}$ es:

$$J = \begin{bmatrix} \frac{\partial x_c}{\partial q_x} & \frac{\partial x_c}{\partial q_f} \\ \frac{\partial z_c}{\partial q_x} & \frac{\partial z_c}{\partial q_f} \end{bmatrix} = \begin{bmatrix} 1 & -\mathrm{L}\sin q_f \\ 0 & \mathrm{L}\cos q_f \end{bmatrix}$$

Y el Jacobiano inverso es:

$$J^{-1} = \begin{bmatrix} \frac{\partial q_x}{\partial x_c} & \frac{\partial q_x}{\partial z_c} \\ \frac{\partial q_f}{\partial x_c} & \frac{\partial q_f}{\partial z_c} \end{bmatrix} = \begin{bmatrix} 1 & \frac{z_c}{\sqrt{\mathrm{L}^2 - z_c^2}} \\ 0 & \frac{L}{\sqrt{\mathrm{L}^2 - z_c^2}} \end{bmatrix}$$

Posicionamiento del codo del robot en superficie plana

Supongamos ahora que deseamos posicionar el punto B. Si aproximamos D $\approx$ L (aproximación razonable, especialmente teniendo en cuenta que el superior radio de la rueda grande compensa aproximadamente la diferencia de longitudes) podemos aplicar como restricciones$q_b = \theta$ y $-q_f = 2\theta$. Expresaremos q_b como variable dependiente de q_f ($q_b = -q_f/2$) y, por lo tanto, nos limitaremos a analizar q_f y q_x. La cinemática directa de la posición del codo será (nótese que $cosx = cos - x$):

$$\vec{r}_B = \begin{bmatrix} x_b \\ z_b \end{bmatrix} = \begin{bmatrix} q_x + \mathrm{D}\cos\frac{q_f}{2} \\ \mathrm{D}\sin\left(-\frac{q_f}{2}\right) \end{bmatrix}$$

El espacio de trabajo es $\{(x_c, z_c) \in \mathbb{R}^2 |\ 0 \leq z_c \leq \mathrm{D}\sin 37{,}5°\}$, ya que los ángulos de track están limitados a 75° por diseño.

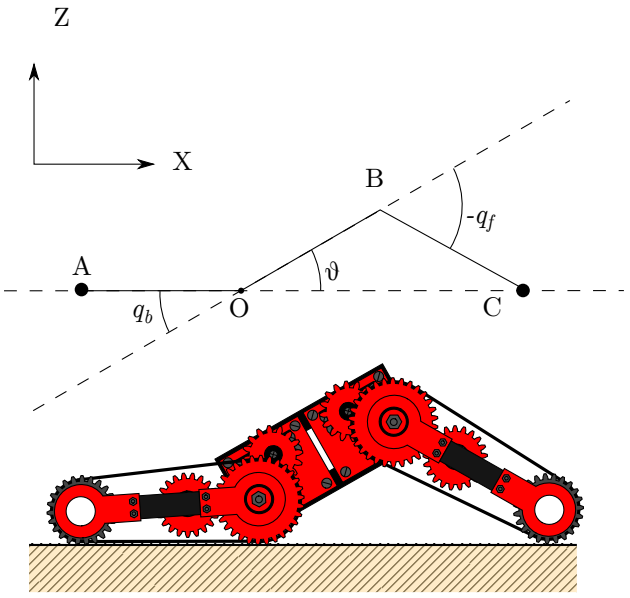

Figura 3.11: Posicionamiento del punto B

La cinemática inversa viene dada por las ecuaciones:

$$q_f = -2 \arcsin \frac{z_b}{\mathrm{D}}$$
$$q_x = x_b - \sqrt{\mathrm{D}^2 - z_b^2}$$

El Jacobiano será:

$$J = \begin{bmatrix} \frac{\partial x_b}{\partial q_x} & \frac{\partial x_b}{\partial q_f} \\ \frac{\partial z_b}{\partial q_x} & \frac{\partial z_b}{\partial q_f} \end{bmatrix} = \begin{bmatrix} 1 & -\frac{\mathrm{D}}{2} \sin q_f \\ 0 & -\frac{\mathrm{D}}{2} \cos\left(-\frac{q_f}{2}\right) \end{bmatrix}$$

Y el Jacobiano inverso:

$$J^{-1} = \begin{bmatrix} \frac{\partial q_x}{\partial x_b} & \frac{\partial q_x}{\partial z_b} \\ \frac{\partial q_f}{\partial x_b} & \frac{\partial q_f}{\partial z_b} \end{bmatrix} = \begin{bmatrix} 1 & \frac{z_b}{\sqrt{\mathrm{D}^2 - z_b^2}} \\ 0 & \frac{-2\mathrm{D}}{\sqrt{\mathrm{D}^2 - z_b^2}} \end{bmatrix}$$

3.3. Control de velocidad de las ruedas en movimiento y maniobras

En la sección anterior hemos supuesto que podemos ajustar q_x libremente mediante los servos modificados, que les proporcionan rotación a las ruedas. Efectivamente, si no existe movimiento en las articulaciones y aplicamos una secuencia de velocidades $\omega(t)$ igual a todas las ruedas, veremos que podemos alcanzar el valor deseado de q_x^* en un tiempo T si:

$$q_x(0) + \int_0^T \omega(t) \cdot R dt = q_x^*$$

No obstante, si existe movimiento articular no podemos garantizar esta condición (incluso si $q_x^* - q_x(0) = 0$) por los siguientes motivos:

- $\omega_{rueda,track}(t)$ es la velocidad angular de la rueda en un sistema de referencia móvil que gira solidariamente con el track.
- La velocidad angular de la rueda respecto a un sistema de referencia fijo es igual a $\omega_{rueda,superficie} = \omega_{rueda,track} + \omega_{track,cuerpo} + \omega_{cuerpo}$, donde $\omega_{track,cuerpo}$ es la velocidad angular del track en el sistema de referencia del robot.
- Si existe movimiento articular, $\omega_{track,cuerpo} + \omega_{cuerpo}$ puede tomar valores distintos de 0 y, por lo tanto, $\omega_{rueda,superficie} \neq \omega_{rueda,track}$ en general.
- Si asignamos a $\omega_{rueda,track}$ un valor de $\dot{q}_x/R$ con la esperanza de que el robot avance a una velocidad $\dot{q}_x$, entonces $\omega_{rueda,superficie} \neq \dot{q}_x/R$ en general.
- Dado que $\omega_{rueda,superficie} = \dot{q}_x/R$ es la condición de no-deslizamiento o condición de rodadura, el hecho de que no se cumpla esta ecuación implica que existe deslizamiento y desconocemos el valor de $\dot{q}_x$, ya que depende de las características dinámicas de los servomotores y de las fuerzas de rozamiento que aparezcan.

Por ello, la velocidad que le proporcionemos a las ruedas depende de $\dot{q}_x$, $\dot{q}_f$ y $\dot{q}_b$. En esta sección calcularemos el valor de la velocidad angular de la rueda en el sistema de referencia del track de acuerdo al modelo cinemático simplificado explicado en la sección 3.2. Distinguiremos dos casos para el cálculo de la velocidad en las ruedas delanteras y otros dos casos para las traseras.

- Cálculo de la velocidad en las ruedas delanteras, rodando sobre la rueda grande.
- Cálculo de la velocidad en las ruedas delanteras, rodando sobre la rueda pequeña.
- Cálculo de la velocidad en las ruedas traseras, rodando sobre la rueda grande.
- Cálculo de la velocidad en las ruedas traseras, rodando sobre la rueda pequeña.

La velocidad del servomotor correspondiente, finalmente, será igual a la velocidad calculada multiplicada por la relación de transmisión invertida:

$$\omega_{servo} = n^{-1}\omega_{rueda,track} = \frac{30}{21}\omega_{rueda,track}$$

La obtención de la expresión para $\dot{\theta}$ (y, dependiendo del caso, para θ) depende de la situación de operación, ya que depende del relieve.

3.3.1. Ruedas delanteras, rodando sobre la rueda grande

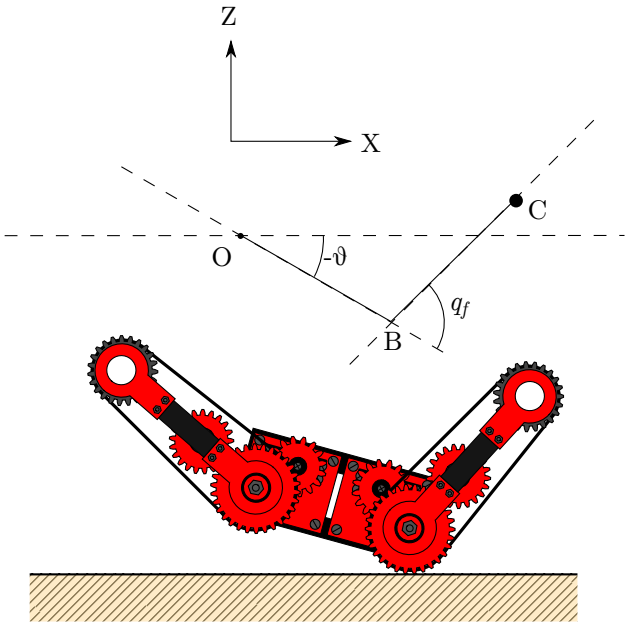

Figura 3.12: Rodando sobre las ruedas delanteras grandes

La velocidad angular de la rueda en el sistema de referencia de la superficie, ω_w, es igual a la velocidad de la rueda sobre el track (ω_R en sentido horario) más la velocidad del track en el sistema de referencia del cuerpo (ω_t) más la velocidad angular del cuerpo (ω_c)

$$\omega_w = -\omega_R + \omega_t + \omega_c$$
$$\omega_w = -\omega_R + \dot{q}_f + \dot{\theta}$$

La velocidad lineal del centro de la rueda es la velocidad del punto O más la velocidad relativa entre ambos puntos:

$$v_B = v_O + v_{OB}$$
$$v_B = \dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta}$$

Aplicando la condición de rodadura, $v_B = -\omega_w R$ (dado que ω_w debe tener sentido horario para coincidir con el sentido del desplazamiento):

$$\dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta} = -R(-\omega_R + \dot{q}_f + \dot{\theta})$$

$$\omega_R = \dot{q}_f + \dot{\theta} + \frac{\dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta}}{R} \tag{3.12}$$

3.3.2. Ruedas delanteras, rodando sobre la rueda pequeña

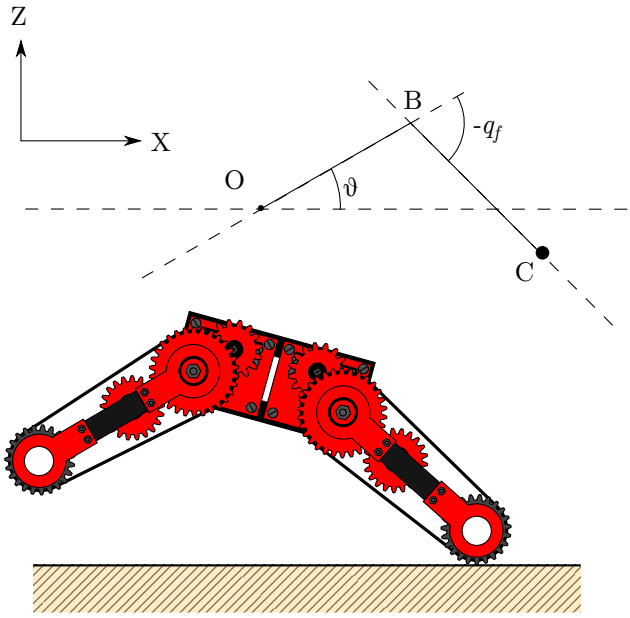

Figura 3.13: Rodando sobre las ruedas delanteras pequeñas

Debido a las correas de transmisión, la velocidad de la rueda pequeña en el sistema de referencia móvil del track es igual a la velocidad de la rueda grande multiplicada por la relación de radios o de dientes: $\omega_r = \frac{R}{r}\omega_R$. La velocidad angular de la rueda en el sistema de referencia de la superficie, ω_w, es igual a la velocidad de la rueda sobre el track (ω_r en sentido horario) más la velocidad del track en el sistema de referencia del cuerpo (ω_t) más la velocidad angular del cuerpo (ω_c)

$$\omega_w = -\omega_r + \omega_t + \omega_c$$
$$\omega_w = -\omega_r + \dot{q}_f + \dot{\theta}$$

La velocidad lineal del centro de la rueda es la velocidad del punto O más la velocidad relativa entre ambos puntos:

$$v_C = v_O + v_{OB}$$
$$v_C = \dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta} - \mathrm{L}\sin(\theta + q_f)(\dot{\theta} + \dot{q}_f)$$

Aplicando la condición de rodadura, $v_C = -\omega_w r$:

$$\dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta} = -r(-\omega_r + \dot{q}_f + \dot{\theta})$$

$$\omega_r = \dot{q}_f + \dot{\theta} + \frac{\dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta} - \mathrm{L}\sin(\theta + q_f)(\dot{\theta} + \dot{q}_f)}{r}$$

$$\omega_R = \frac{r}{R}\left(\dot{q}_f + \dot{\theta}\right) + \frac{\dot{q}_x - \mathrm{D}\sin(\theta)\dot{\theta} - \mathrm{L}\sin(\theta + q_f)(\dot{\theta} + \dot{q}_f)}{R} \tag{3.13}$$

3.3.3. Ruedas traseras, rodando sobre las ruedas grandes

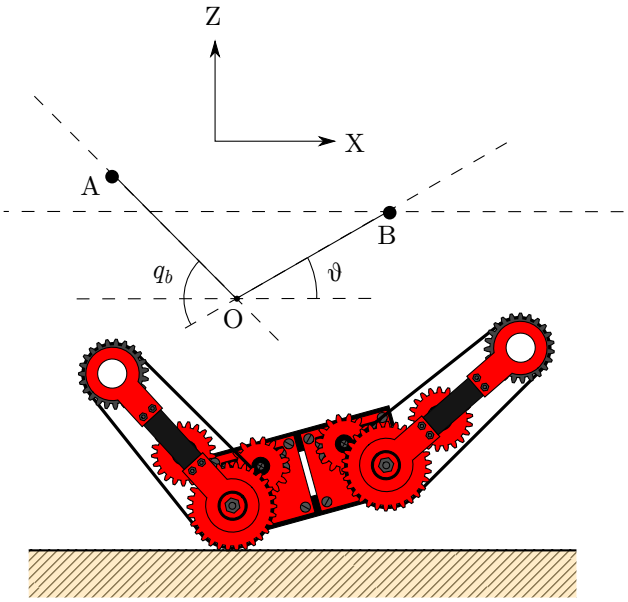

Figura 3.14: Rodando sobre las ruedas traseras grandes

La velocidad angular de la rueda en el sistema de referencia de la superficie, ω_w, es igual a la velocidad de la rueda sobre el track (ω_R en sentido horario) más la velocidad del track en el sistema de referencia del cuerpo (ω_t) más la velocidad angular del cuerpo (ω_c)

$$\omega_w = -\omega_R + \omega_t + \omega_c$$
$$\omega_w = -\omega_R - \dot{q}_b + \dot{\theta}$$

La velocidad lineal del centro de la rueda es la velocidad del punto O:

$$v_O = \dot{q}_x$$

Aplicando la condición de rodadura, $v_O = -\omega_w R$:

$$\dot{q}_x = -R(-\omega_R - \dot{q}_b + \dot{\theta})$$

$$\omega_R = -\dot{q}_b + \dot{\theta} + \frac{\dot{q}_x}{R} \tag{3.14}$$

3.3.4. Ruedas traseras, rodando sobre las ruedas pequeñas

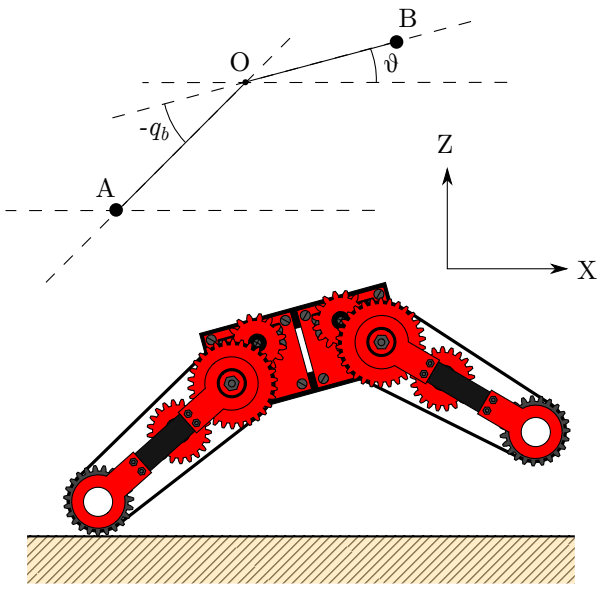

Figura 3.15: Rodando sobre las ruedas traseras pequeñas

Debido a las correas de transmisión, la velocidad de la rueda pequeña en el sistema de referencia móvil del track es igual a la velocidad de la rueda grande multiplicada por la relación de radios o de dientes: $\omega_r = \frac{R}{r}\omega_R$. La velocidad angular de la rueda en el sistema de referencia de la superficie, ω_w, es igual a la velocidad de la rueda sobre el track (ω_R en sentido horario) más la velocidad del track en el sistema de referencia del cuerpo (ω_t) más la velocidad angular del cuerpo (ω_c)

$$\omega_w = -\omega_R + \omega_t + \omega_c$$
$$\omega_w = -\omega_R - \dot{q}_b + \dot{\theta}$$

La velocidad lineal del centro de la rueda es la velocidad del punto O más la velocidad relativa entre ambos puntos:

$$v_A = v_O + v_{OA}$$
$$v_A = \dot{q}_x + \mathrm{L}\sin(\theta - q_b)(\dot{\theta} - \dot{q}_b)$$

Aplicando la condición de rodadura, $v_A = -\omega_w r$:

$$\dot{q}_x + \mathrm{L}\sin(\theta - q_b)(\dot{\theta} - \dot{q}_b) = -r(-\omega_r - \dot{q}_b + \dot{\theta})$$

$$\omega_r = -\dot{q}_b + \dot{\theta} + \frac{\dot{q}_x + \mathrm{L}\sin(\theta - q_b)(\dot{\theta} - \dot{q}_b)}{r}$$

$$\omega_R = \frac{r}{R}\left(\dot{\theta} - \dot{q}_b\right) + \frac{\dot{q}_x + \mathrm{L}\sin(\theta - q_b)(\dot{\theta} - \dot{q}_b)}{R} \tag{3.15}$$

Capítulo 4

Descripción del sistema electrónico

En los robots y, en general, en las máquinas y vehículos automatizados, la electrónica de control cumple el papel que el sistema nervioso desempeña en los seres vivos. El sistema electrónico conecta los sensores, que recopilan información del medio, con los dispositivos de control, que gestionan esa información y evalúan las acciones a llevar a cabo, y los actuadores, que llevan a cabo las operaciones determinadas por las unidades de control. Adicionalmente, la electrónica de potencia se encarga de suministrar la energía para que todos estos elementos puedan llevar a cabo su función.

Este capítulo está estructurado de la siguiente forma:

- La sección 4.1 presenta el esquema general del sistema
- La sección 4.2 describe los sistemas de potencia que alimentan los diferentes subsistemas
- La sección 4.3 explica las características de los sensores empleados
- La sección 4.4 describe la unidad de control
- La sección 4.5 contiene una descripción de los motores y su control

4.1. Descripción general

Los principales elementos del sistema son:

- Una batería de 2200 mAh y 7.4 V, un conversor continua-continua con salidas de 5 V y 6 V y un regulador de tensión LM317 acondicionado para proporcionar una salida de 3V
- Dos sensores de proximidad basados en infrarrojos y un acelerómetro de 3 ejes
- Un microcontrolador Arduino Mega2560 conectado por USB a un PC
- 4 servomotores Futaba S3010 y 4 servomotores Futaba S3003 modificados para rotación continua

La figura 4.1 muestra el esquema general de la electrónica del robot

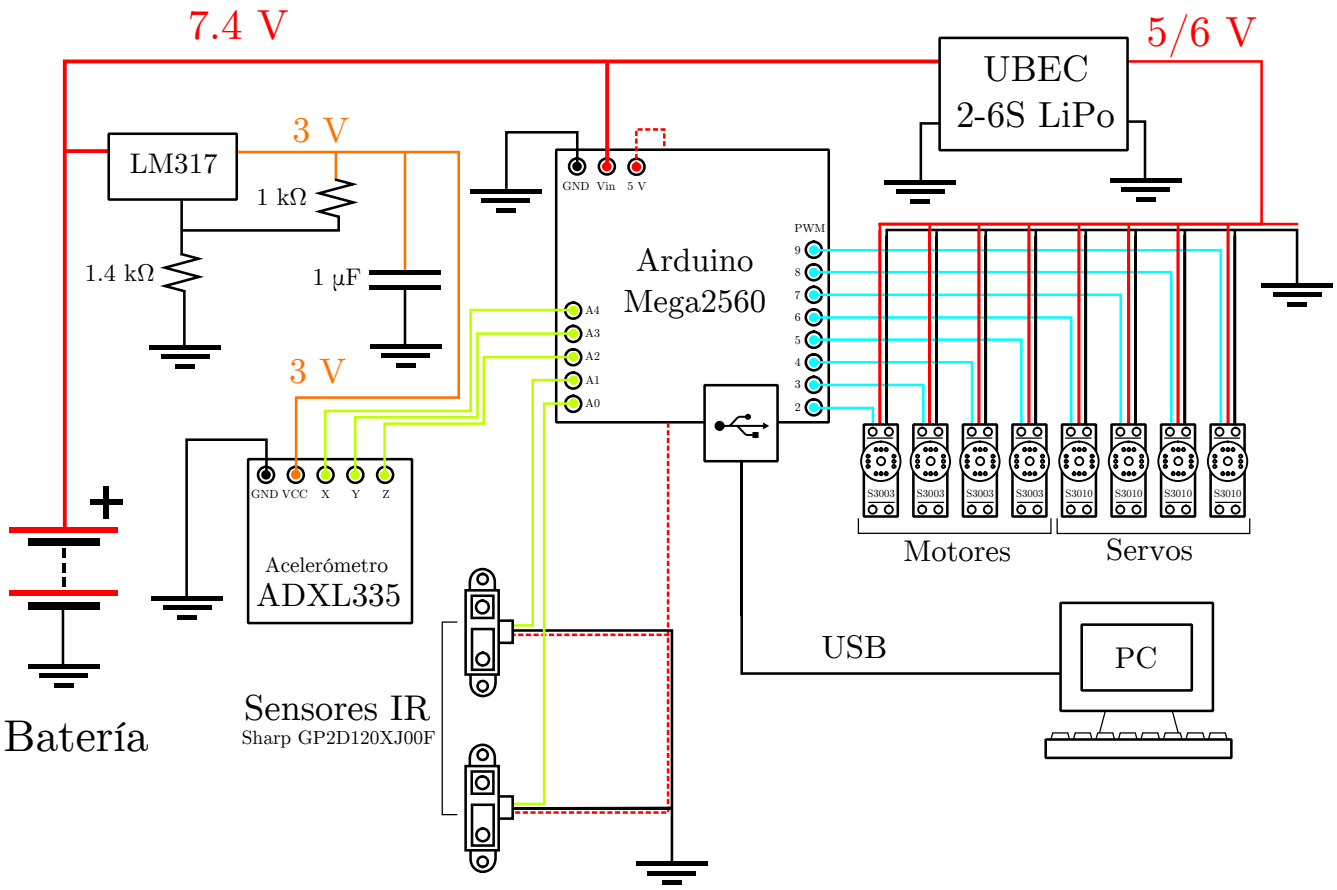

Figura 4.1: Esquema de la electrónica del robot

4.2. Etapas de potencia

Toda la alimentación de los subsistemas del robot parte de la batería. Sin embargo, cada subsistema del robot es alimentado de una forma diferente:

- El microcontrolador Arduino Mega2560 se alimenta directamente de la batería
- Los servos y motores se alimentan de la salida de un conversor de continua a continua a 5 V o 6 V (seleccionado mediante un jumper). La configuración empleada es 5 V, pero los motores también pueden operar a 6 V
- El acelerómetro ADXL335 no soporta tensiones de alimentación superiores a 3.6 V, por lo que es alimentado mediante un regulador de tensión (LM317), acondicionado para proporcionar esa tensión
- Los sensores infrarrojos son alimentados directamente desde el pin de 5 V del Arduino.

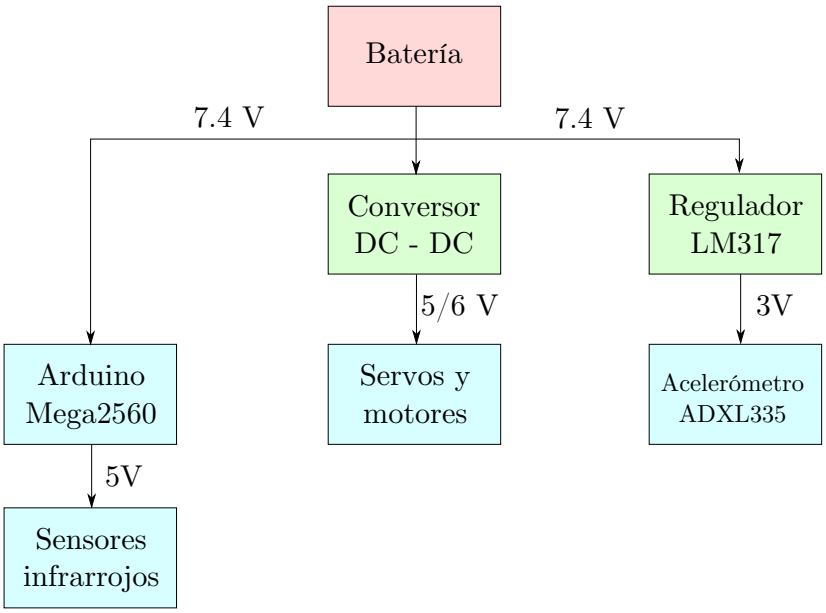

Figura 4.2: Esquema de la alimentación de los subsistemas

4.2.1. Batería

La fuente de energía del robot es una batería recargable Turnigy 2.2 (2200 mAh de capacidad) de dos células y una tasa de descarga de 20C (pico de 30C). Proporciona suficiente corriente para alimentar todos los sistemas del robot, ya que la corriente de descarga máxima es de 44 A.

Figura 4.3: Batería

4.2.2. Conversor DC - DC

Ambos modelos de servomotor requieren alimentación a una tensión de entre 4.8 V y 6V. Por ello, alimentarlos directamente desde la bateria sería dañino. Emplearemos un conversor UBEC 2-6S LiPo para alimentarlos. Este conversor se alimenta a tensiones de entre 5.5v y 26v (por lo que podemos alimentarlo con la batería), proporciona una corriente de salida de 3 A (lo que es suficiente para alimentar los servomotores en el caso más desfavorable), y una salida de 5 V o 6 V seleccionable mediante un jumper.

En este proyecto, se ha seleccionado la salida a 5 V puesto que de esta forma coincide con el valor alto de las salidas digitales del microcontrolador, ya que el control de los servos se realiza mediante señales PWM directamente desde el Arduino Mega2560.

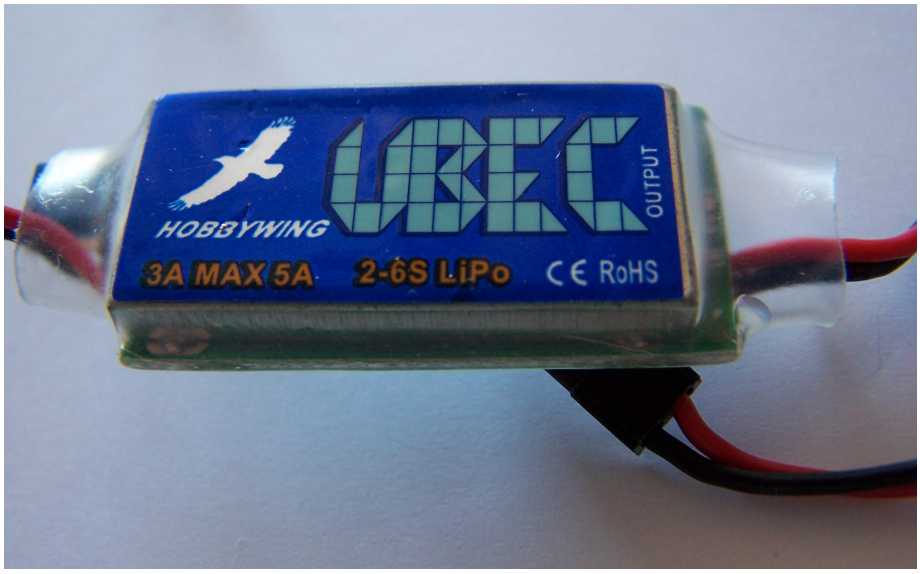

Figura 4.4: Conversor continua - continua

4.2.3. Circuito de alimentación del acelerómetro

Para la alimentación del acelerómetro emplearemos una solución de potencia más reducida que en el caso de los servos, ya que el ADXL335 requiere una corriente de unos 350 µA. El ADXL335 opera a tensiones de entrada de entre 1.8 V y 3.6 V, por lo que es necesario crear un circuito de potencia con esas características. El circuito diseñado se basa en el integrado LM317, un regulador lineal que puede emplearse como regulador variable de corriente y de tensión. La figura 4.5 ilustra el acondicionamiento habitual para un LM317 como regulador de tensión:

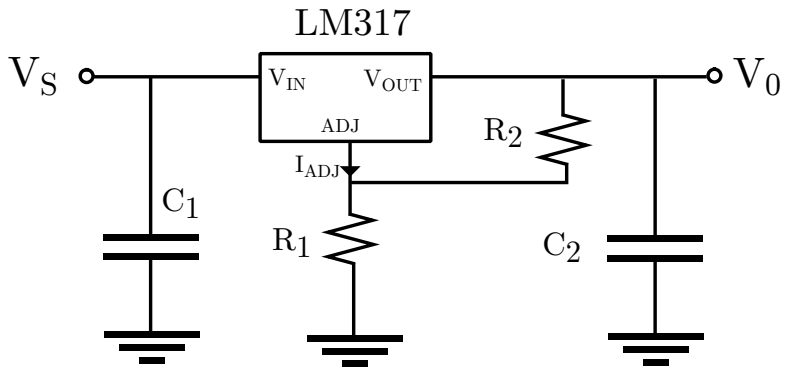

Figura 4.5: Acondicionamiento del LM317 como regulador de tensión

Con este esquema, la tensión de salida (en voltios) viene dada por la expresión:

$$V_0 = 1{,}25\left(1 + \frac{R_2}{R_1}\right) + I_{ADJ}R_2$$

Siendo el término $I_{ADJ}R_2$ despreciable para la mayoría de los propósitos. En nuestro caso, escogeremos los valores $R_1 = 1$ kΩ y $R_2 = 1{,}4$ kΩ, de modo que $V_0 \approx 3$ V.

En cuanto a los condensadores, el condensador C_2 mejora la respuesta transitoria del circuito, mientras que C_1 se emplea cuando el circuito está físicamente lejano a la fuente de tensión (por lo que no se ha instalado en este proyecto)

4.2.4. Alimentación del microcontrolador y los sensores infrarrojos

El microcontrolador, un Arduino Mega2560, tiene como límites de la tensión de entrada 6 V (inferior) y 20 V (superior). No obstante, el rango recomendado de alimentación es de entre 7 y 12 V, ya que tensiones inferiores a 7 V podrían no proporcionar suficiente estabilidad a la placa y tensiones superiores a 12 V podrian causar daños en algunos componentes. Por ello, alimentaremos el microcontrolador directamente desde la batería, a 7V.

En la placa del microcontrolador existe un pin de 5 V del que podemos alimentar directamente los sensores infrarrojos, ya que requieren una alimentación a entre 4.5 y 5.5 V pero no presentan los elevados requisitos de potencia de los servos. Esta tensión procede del regulador on-board del Arduino y alimenta la mayoría de los dispositivos de este.

4.3. Sistema de percepción

El sistema de percepción consta de dos elementos:

- Los sensores de proximidad por infrarrojos de corto alcance Sharp GP2D120XJ00F
- Un acelerómetro de 3 ejes ADXL335.

4.3.1. Sensores de proximidad

Para detectar obstáculos y reconocer el terreno, el robot emplea dos sensores de proximidad por infrarrojos Sharp GP2D120XJ00F. Estos sensores son útiles para reconocer distancias de entre 4 y 30 cm. Estos sensores proporcionan una salida analógica de entre 0 y aproximadamente 3.1 V en función de la distancia.

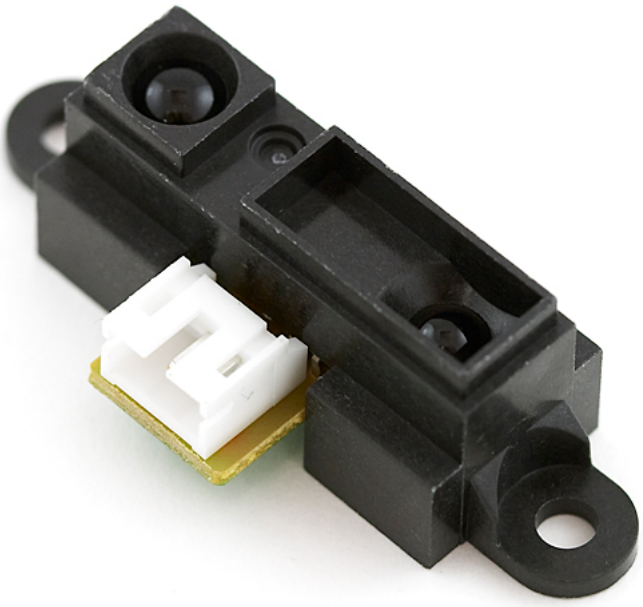

Figura 4.6: Sharp GP2D120XJ00F

Estos sensores se ubican en los tracks delanteros, "mirando"hacia adelante y a unos 3 cm de la punta del robot (detrás, no delante). El motivo detrás de la decisión de ubicarlos en ese lugar es la naturaleza de la función de respuesta del sensor. En la datasheet del sensor se nos proporciona dicha información, que reproducimos en la figura 4.7:

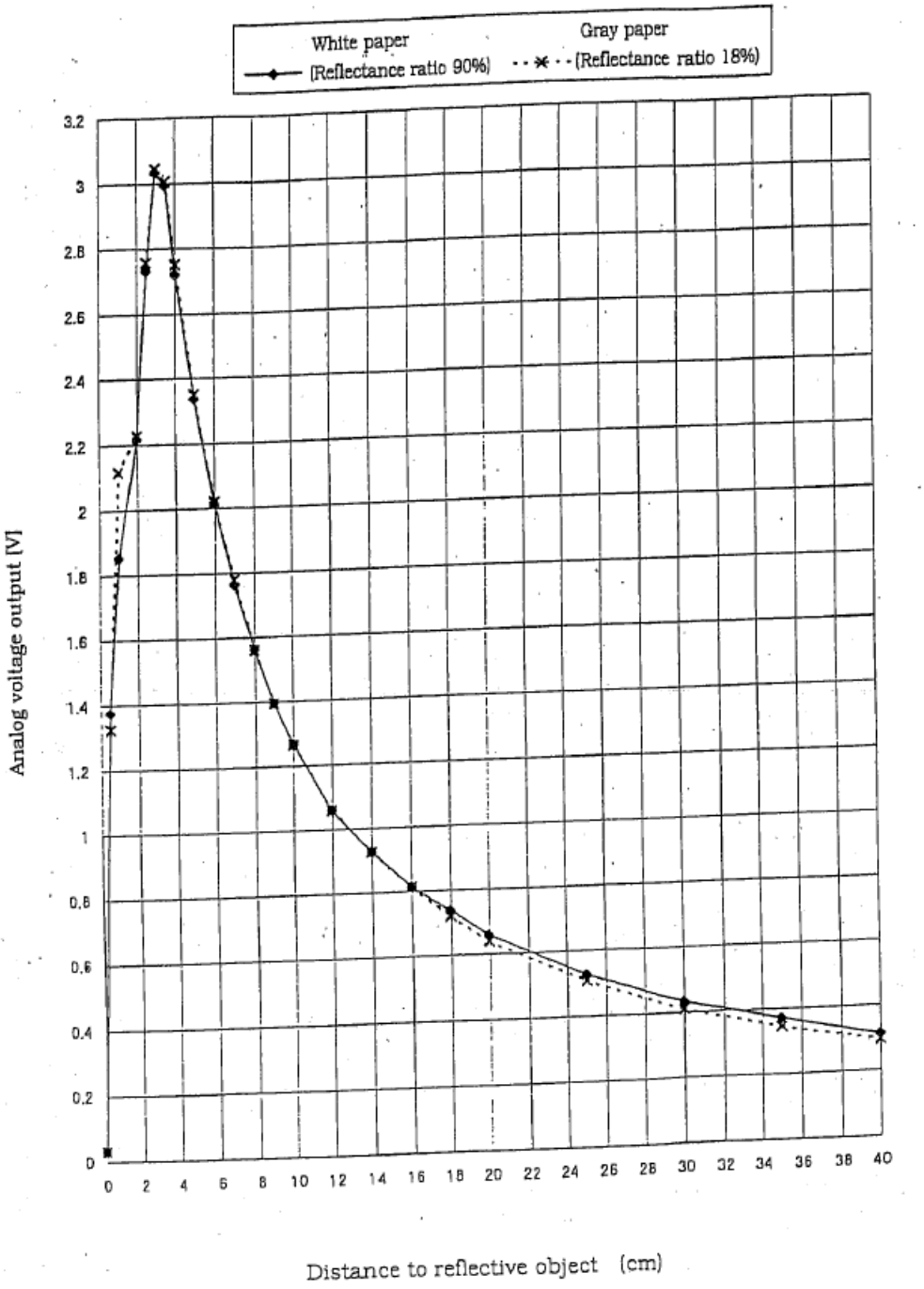

Figura 4.7: Característica de respuesta del sensor

Como se puede observar en la figura, la curva de respuesta del sensor no es una función inyectiva. Ello tiene implicaciones relevantes, puesto que una medición del voltaje puede corresponder a varios valores diferentes de distancia, y diferenciar entre ellos puede ser una tarea no trivial. Sin embargo, si desplazamos el sensor respecto al extremo del robot de modo que se encuentre a 3 cm detrás de la punta, podemos realizar un cambio de variables e ignorar la parte de la curva a la izquierda de los 3 cm, obteniendo de esta forma una curva monotónicamente decreciente que no presenta el problema que aparecería si colocáramos el sensor exactamente en la punta del track.

Calibración del sensor

Para emplear el sensor, aproximaremos la curva de respuesta mediante una función matemática. En la datasheet se proporciona la gráfica de la figura 4.8, en la que se observa que, transformando una variable, se puede obtener una función que es prácticamente lineal en gran parte de su dominio.

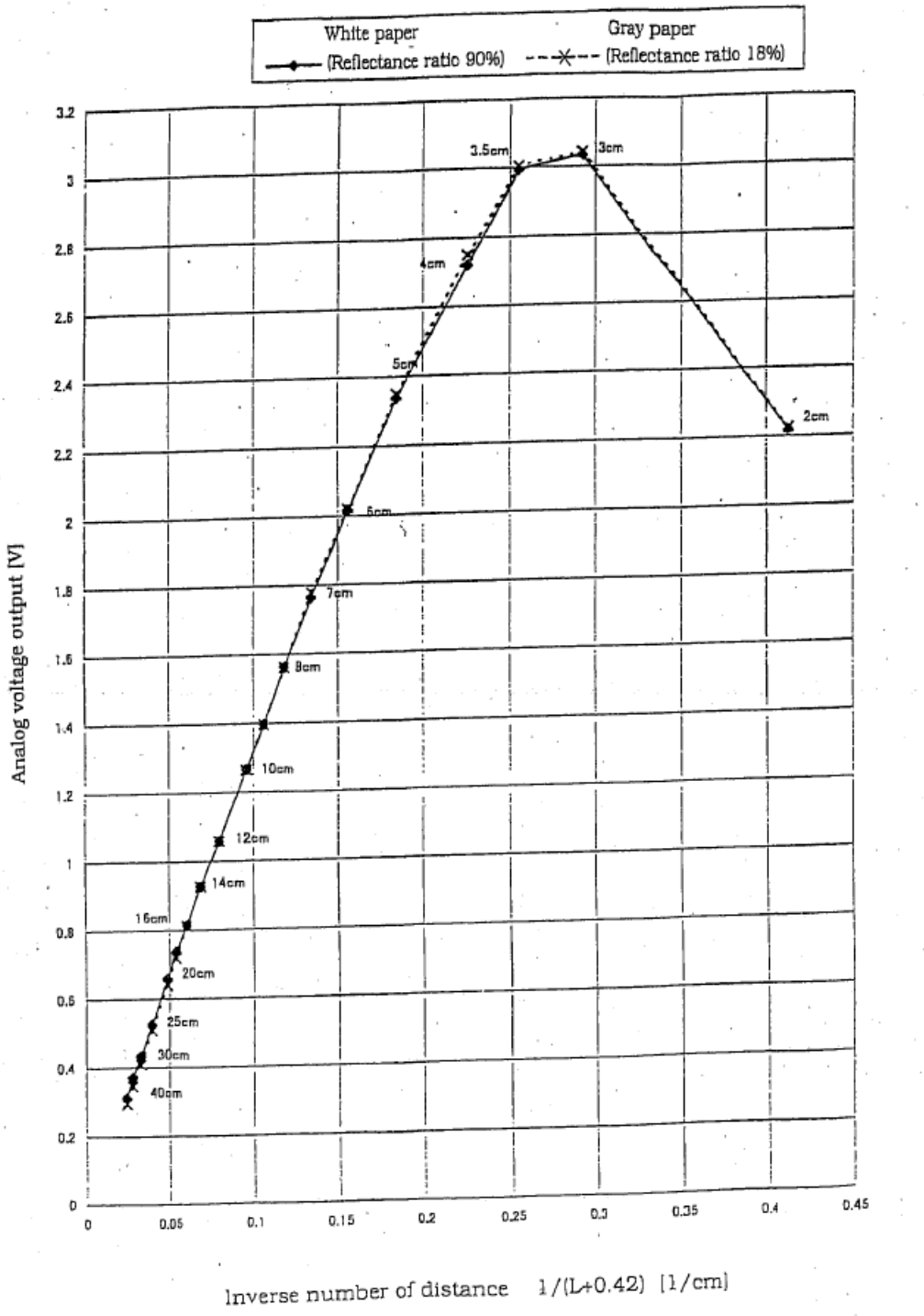

Figura 4.8: Característica de respuesta del sensor

Dado que también nos interesa la parte no lineal de la gráfica hasta aproximadamente 3 cm, una aproximación lineal podría no resultar conveniente en estas distancias. Con el fin de emplear una única aproximación para todo el rango relevante, ajustaremos un polinomio de segundo grado a un conjunto de datos obtenidos empíricamente para compararlo a continuación con los datos de la datasheet. Estimaremos una función de la forma:

$$V(L) = \beta_0 + \beta_1 \frac{1}{L + 0{,}42} + \beta_2 \frac{1}{(L + 0{,}42)^2}$$

donde la distancia al sensor, L, es la distancia a la punta del track más 3 centímetros: $L = d + 3$. Por lo tanto, podemos reescribir la función como:

$$V(d) = \beta_0 + \beta_1 \frac{1}{d + 3{,}42} + \beta_2 \frac{1}{(d + 3{,}42)^2}$$

Los datos se presentan en la tabla 4.1

Distancia (cm)	Tensión (V)
0.0	2.942
0.5	2.878
1.0	2.693
1.5	2.575
2.0	2.277
2.5	2.038
3.0	1.964
3.5	1.798
4.0	1.671
4.5	1.593
5.0	1.505
5.5	1.417
6.0	1.353
6.5	1.251
7.0	1.177
7.5	1.114
8.0	1.055
8.5	0.997
9.0	0.943
9.5	0.894
10.0	0.865

Cuadro 4.1: Datos de calibración del sensor

Los valores estimados para los parámetros se presentan en la tabla 4.2

Parámetro	Valor	Unidad
β_0	-0.543	V
β_1	20.77	V·cm
β_2	-29.516	V·cm^2

Cuadro 4.2: Datos de calibración del sensor

En la figura 4.9 se puede observar el ajuste y la comparación con la gráfica de la datasheet.

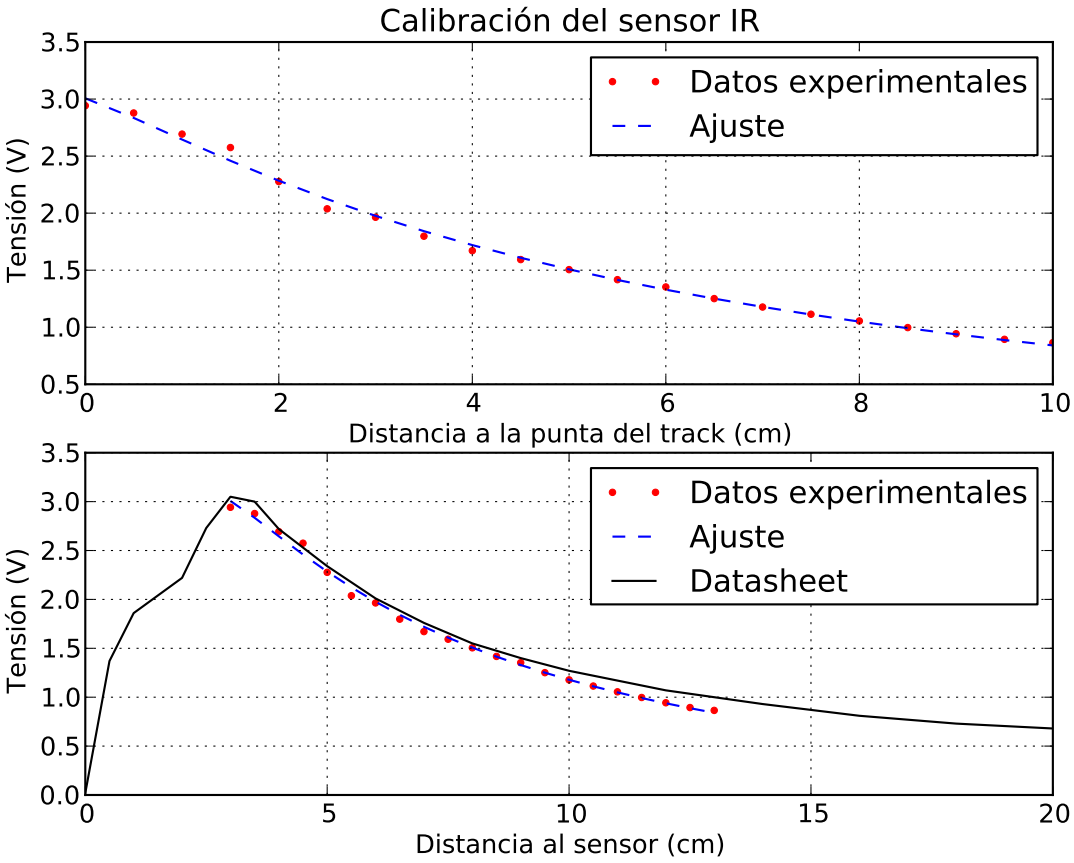

Figura 4.9: Curva de respuesta estimada

Se puede observar que el ajuste es válido, especialmente teniendo en cuenta un error de ± 1 mm en las mediciones. Los datos experimentales difieren moderadamente de los de la datasheet, discrepancia atribuible a los siguientes factores:

- Diferencias entre la tensión de alimentación
- Diferencias en las características ópticas del material reflectante
- Existencia de impedancias de salida significativas debidas a la longitud del cable de conexión.

Por lo tanto, la expresión de la curva de respuesta es finalmente:

$$V(d) = \beta_0 + \beta_1 \frac{1}{d + 3{,}42} + \beta_2 \frac{1}{(d + 3{,}42)^2}$$

$$d(V) = \frac{2\beta_2}{\sqrt{\beta_1^2 - 4\beta_2(\beta_0 - V)} - \beta_1} - 3{,}42$$

4.3.2. Acelerómetro

Como acelerómetro emplearemos el ADXL335. Se trata de un acelerómetro analógico de tres ejes de bajo ruido y bajo consumo, con un rango de medida ±3g. El formato adquirido incluye condensadores de 0.1 µF montados en la placa para limitar el ancho de banda de acada eje a 50 Hz.

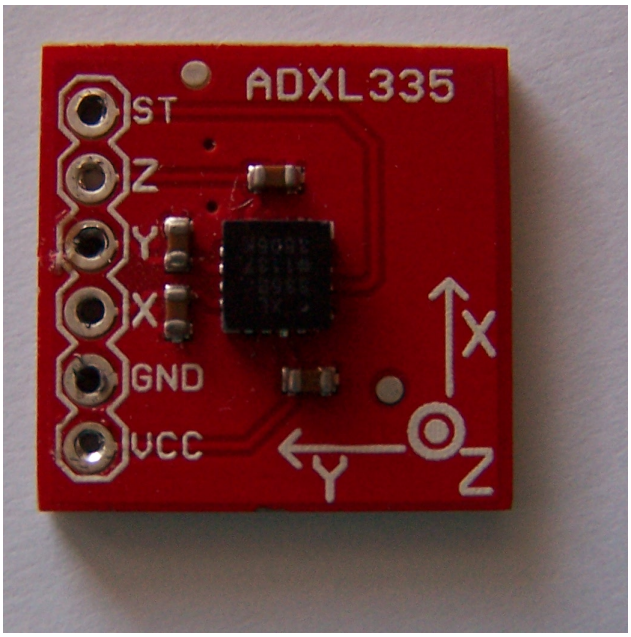

Figura 4.10: Acelerómetro ADXL335

Se trata de un sensor analógico, en el que la tensión de cada uno de los pines correspondientes responde linealmente con la aceleración en el mismo eje:

$$V_i(a_i) = V_{0,i} + S_i a_i$$

donde S_i es la sensibilidad en el eje. Según la datasheet, los valores típicos son un offset V_0 de en torno a 1.5 V y una sensibilidad S de unos 300 mV/g (donde g = 9.8 m·s$^-$1); dado que estos valores pueden diferir, los estimaremos empíricamente tomando medidas colocando el acelerómetro en posiciones ortogonales. Los resultados se muestran en la tabla 4.3

Eje	Offset V_0 (V)	Sensibilidad (V/g)
x	1.643	0.331
y	1.650	0.350
z	1.700	0.300

Cuadro 4.3: Calibración del acelerómetro

Cálculo de la inclinación

La principal utilidad del acelerómetro es determinar la inclinación del cuerpo del robot. Si el robot no está acelerando ni rotando, la única aceleración medida será la proyección de la gravedad sobre los ejes del sistema de referencia del robot. Los motivos por los que esta estimación es fiable son que el robot está parado o en un movimiento lineal la mayor parte del tiempo y, que cuando el robot sufre aceleraciones y rotaciones, éstas son de magnitud reducida comparadas con la gravedad. Adicionalmente, en la mayoría de los casos se pueden detectar la presencia (si bien no el valor concreto) de aceleraciones adicionales e ignorar la medida (fundamentalmente, cuando $a_x^2 + a_y^2 + a_z^2$ tiene un valor significativamente distinto de g^2)

El sensor queda fijado a la estructura del robot de modo que el eje x queda alineado con el tronco o cuerpo del robot, con el sentido positivo apuntando hacia el frontal del robot. La figura 4.11 ilustra los sistemas de referencia escogidos y los ángulos que describen la orientación del vehículo respecto al sistema de referencia de la superficie.

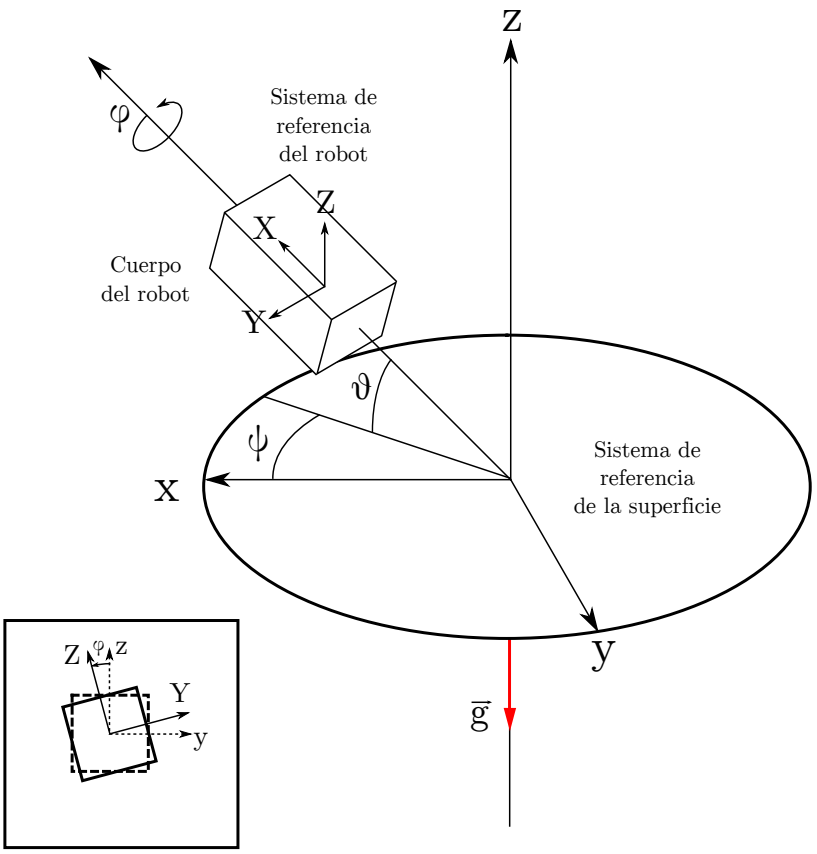

Figura 4.11: Sistemas de referencia

Mediante un acelerómetro y empleando únicamente la gravedad como referencia, no es posible medir el ángulo de guiñada ψ, ya que la gravedad es un vector normal al plano xy del sistema de referencia fijo y las proyecciones de la gravedad sobre el sistema de referencia móvil no dependen de este ángulo. Sin embargo, sí podemos medir los ángulos de cabeceo θ y de alabeo ϕ. En lo sucesivo, asumimos que $\psi = 0$ para simplificar los cálculos.

En la convención empleada, la orientación del sistema de referencia móvil es el producto de aplicar a un sistema de referencia de la misma orientación que el fijo las transformaciones:

1. Una rotación ϕ en torno al eje x del sistema de referencia fijo.
2. Una rotación θ en torno al eje y del sistema de referencia fijo.
3. Una rotación ψ en torno al eje z del sistema de referencia fijo (que, de nuevo, asumimos nula).

Por lo tanto, la matriz de la transformación será el producto de las matrices de rotación correspondientes a estas transformaciones, teniendo en cuenta el sentido de los ángulos indicado en la figura 4.11:

$$T(\theta,\phi) = R_z(\psi=0)R_y(\theta)R_x(\phi) = IR_y(\theta)R_x(\phi) = R_y(\theta)R_x(\phi) =$$

$$= \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & \sin\phi \\ -\sin\theta & -\sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

Dado que la gravedad es $\vec{g} = -g \cdot \hat{e}_z$, podemos calcular la expresión de la gravedad en el sistema de referencia móvil como:

$$T(\theta,\phi)\cdot(-g\cdot\hat{e}_z) = \begin{bmatrix} \cos\theta & -\sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & \sin\phi \\ -\sin\theta & -\sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} = \begin{bmatrix} -\sin\theta\cos\phi \\ -\sin\phi \\ -\cos\phi\cos\theta \end{bmatrix} g$$

$$a_x(\theta,\phi) = -g\sin\theta\cos\phi$$

$$a_y(\theta,\phi) = -g\sin\phi$$

$$a_z(\theta,\phi) = -g\cos\theta\cos\phi$$

Por lo tanto, podemos calcular los ángulos de cabeceo y alabeo resolviendo el sistema de ecuaciones anterior:

$$\theta = -\text{atan2}(a_x, a_z)$$

$$\phi = \arcsin\left(\frac{-a_y}{\sqrt{a_x^2 + a_y^2 + a_z^2}}\right)$$

Donde la función $\text{atan2}(y,x)$ es el ángulo entre el sentido positivo del eje x y el vector de coordenadas (x, y). Es similar a la función $\arctan(y/x)$; sin embargo, corrige el resultado dependiendo del cuadrante del plano en el que se halla el punto, ya que la función arcotangente no distingue entre direcciones diametralmente opuestas. Esta función se encuentra en la mayoría de los lenguajes de programación relevantes en ciencia e ingeniería, incluyendo, entre muchos otros, FORTRAN, la librería estándar de C `math.h` y Python.

4.4. Unidad de control

La unidad de procesamiento está compuesta por un Arduino Mega2560, revisión 3. Destacamos las siguientes características de este microcontrolador:

- Alimentación a entre 7 y 12 V mediante un jack o el pin `Vin`.
- 54 pines de entrada/salida digitales, de los cuales 14 pueden ser empleados como salidas PWM, 8 de ellos pueden emplearse como 4 puertos serie por hardware (2 pines por puerto, `TX` y `RX`), y 6 de ellos pueden ser configurados para interrupciones externas.
- 16 entradas analógicas.
- Microprocesador ATmega2560 a 16 MHz.
- 256 KB de memoria Flash (248 KB usables), 8 KB de memoria SRAM y 4 KB de memoria EEPROM
- Conversor serie a USB integrado para el primer puerto serie por hardware, basado en un ATMega16U2.
- Es hardware open-source.

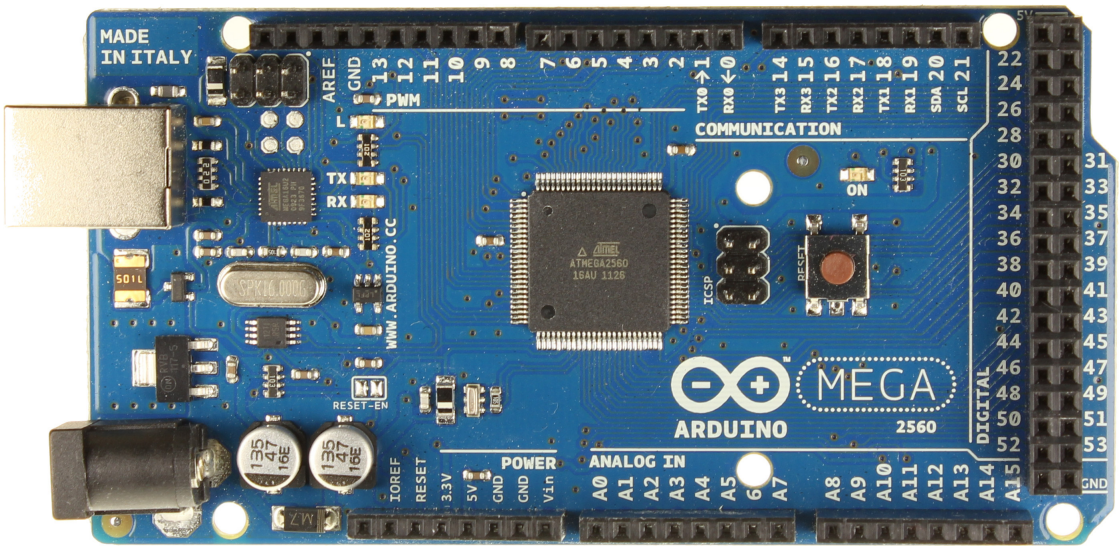

Figura 4.12: Arduino Mega2560 rev3

Las principales ventajas que tiene el Arduino Mega2560 respecto a otros microcontroladores Arduino son la mayor cantidad de entradas y salidas y un microprocesador de una capacidad superior. Para este proyecto, poseer al menos 8 salidas PWM y 5 entradas analógicas permite implementar la electrónica de control con un único microcontrolador, además de proporcionar una gran capacidad de expansión.

La programación del Arduino Mega2560 se describe en el capítulo 5 y, por lo tanto, no se hará énfasis aquí en ella.

4.5. Sistema motriz

Como se ha descrito en el capítulo 3, el sistema motriz del robot está compuesto por dos tipos de servomotores:

- Los Futaba S3003 modificados para rotación continua mueven las ruedas.
- Los Futaba S3010 levantan los tracks.

4.5.1. Control en posición

Electrónicamente, los Futaba S3010 (al igual que los S3003 si no hubieran sido modificados) se controlan mediante una señal PWM de 50 Hz de frecuencia (20ms de periodo). Variando el ciclo de trabajo es posible controlar la posición del servomotor, de forma que una anchura de pulso de 0.3 ms lo posiciona en un extremo, una anchura de 2.3 ms lo posiciona en el extremo contrario y una anchura intermedia lo posiciona en un lugar intermedio, siendo la relación lineal:

$$\theta(T) = \frac{90^\circ}{1\text{ ms}}(T - 1{,}3\text{ ms})$$

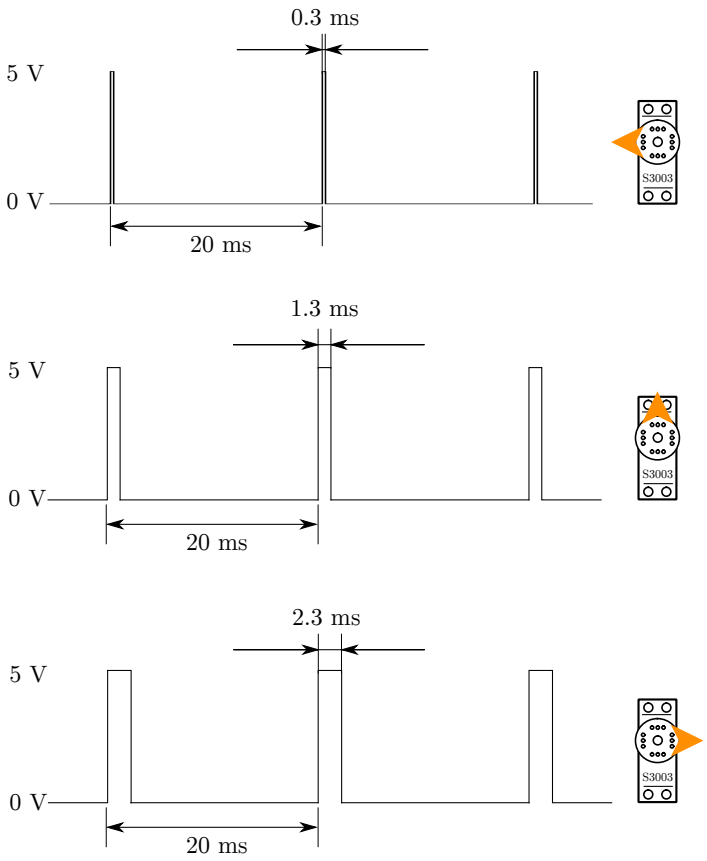

Figura 4.13: Control PWM de los servos

4.5.2. Control en velocidad

Los servos Futaba S3003 han sido modificados para rotar continuamente en lugar de ser controlados en velocidad. El control en velocidad es similar al control en posición, siendo las dos principales diferencias:

- Al no existir un lazo de control cerrado, la aparición de pares resistentes modifica la relación entre la señal de control y la velocidad
- La relación entre el ciclo de trabajo y la velocidad no es lineal

La figura 4.14 ilustra de forma gráfica la forma de la relación de control.

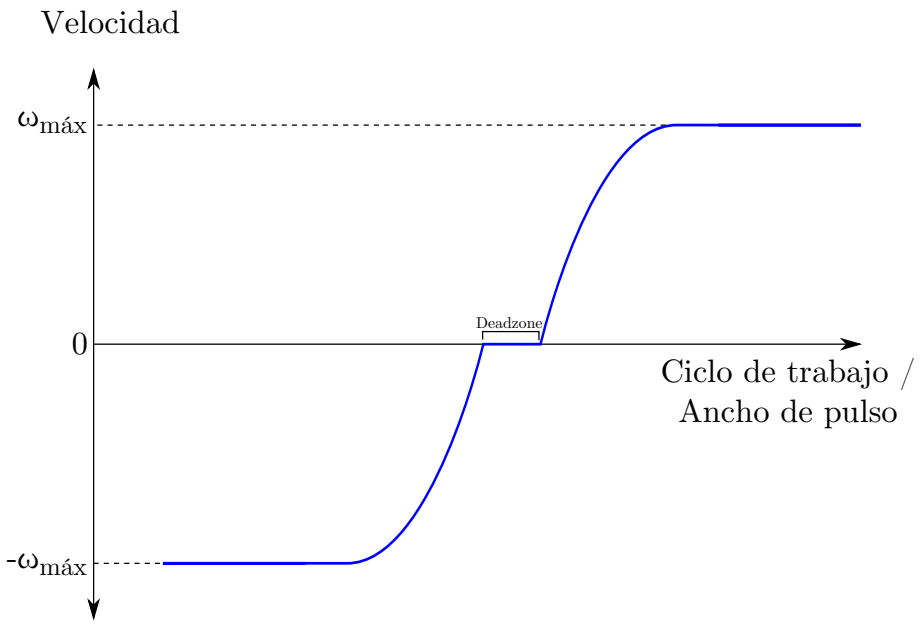

Figura 4.14: Control PWM en velocidad de los servos

Para caracterizar dicha función, se ha medido de forma empírica el tiempo que tardaba el servo en dar un número entero de vueltas para diferentes valores del ancho de pulso y se ha ajustado una función a trozos compuesta por las siguientes partes (en cada mitad, ya que la función tiene simetría impar en torno al punto medio):

- Una constante (recta horizontal)
- Un polinomio de segundo grado tangente a la constante
- Un polinomio de primer grado (recta) tangente al polinomio de segundo grado
- Una zona muerta (constante 0)

Los resultados de este ajuste se pueden observar en la figura 4.15

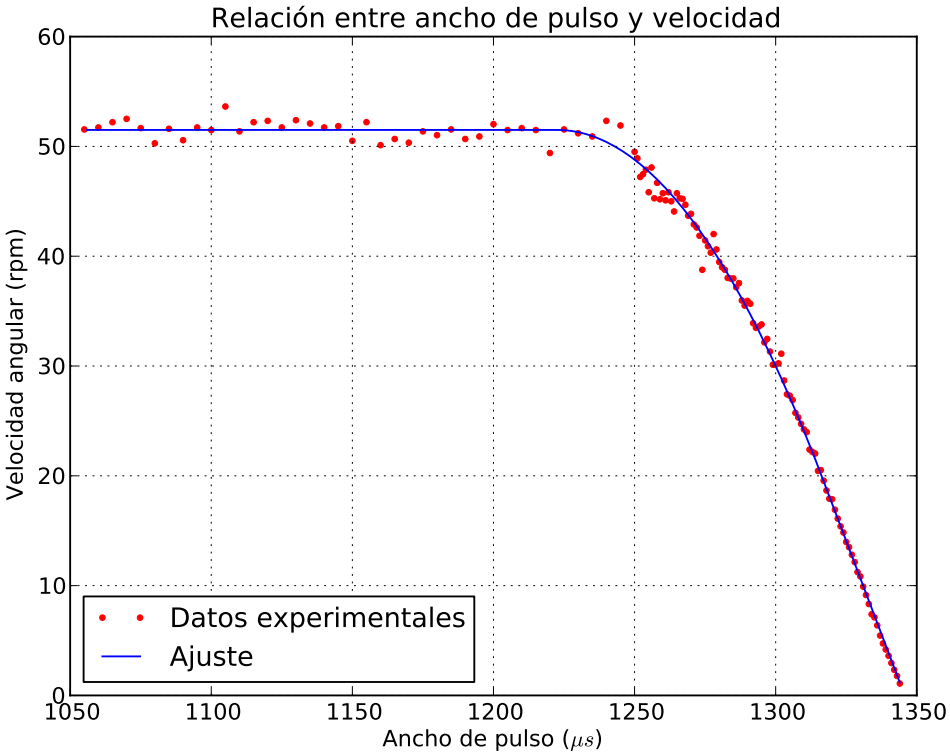

Figura 4.15: Función ajustada

Debe tenerse en cuenta que la ubicación exacta de la zona muerta y, por lo tanto, de todo la función, depende del valor del divisor resistivo que sustituye al potenciómetro, valor que puede presentar variaciones entre los servos. Por ello, se debe determinar experimentalmente en cada servo modificado. Los valores obtenidos se presentan en la tabla 4.4

Track del servo	Comienzo (µs)	Final (µs)
Frontal derecho	1475	1481
Frontal izquierdo	1344	1350
Trasero derecho	1430	1436
Trasero izquierdo	1351	1357

Cuadro 4.4: Zona muerta en los S3003 modificados

Estos valores son tenidos en cuenta por el software, como se explica en el capítulo correspondiente.

Capítulo 5

Descripción del software

El firmware y el software cargados en la unidad de control contienen la lógica de operación de todo robot. Si bien es posible delegar ciertas funciones en la electrónica (por ejemplo, si se emplea un circuito de acondionamiento del sensor), el software generalmente implementa la mayoría de las funciones del robot, tanto de percepción como de decisión y actuación.

En este capítulo podemos encontrar las siguientes secciones:

- La sección 5.1 describe el protocolo de comunicación especificado y el esquema de comunicación entre el microcontrolador y el PC
- La sección 5.2 describe el software del microcontrolador
- La sección 5.3 describe el software del PC

5.1. Comunicaciones

5.1.1. Esquema de comunicaciones

La figura 5.1 muestra el esquema de comunicaciones entre el microcontrolador y el PC:

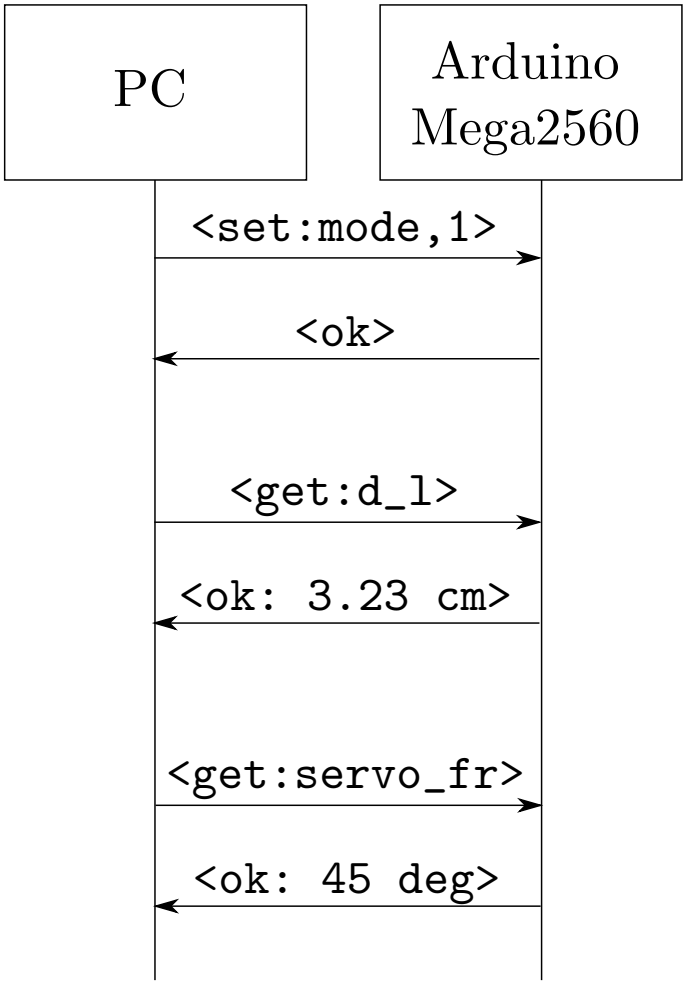

Figura 5.1: Comunicación

Cada ciclo de comunicación se inicia con el software del PC enviando un mensaje (que denominaremos pregunta) al microcontrolador. El PC esperará a que el microcontrolador devuelva un mensaje (denominado respuesta) antes de enviar otro mensaje. De este modo se garantiza que no se acumulan datos en el puerto sin leer, lo que puede ocasionar la pérdida de datos, y a la vez se confirma que el microcontrolador ha recibido un mensaje correctamente formateado, ya que en este caso el microcontrolador puede avisar de ello en el mensaje de respuesta.

5.1.2. Protocolo de comunicaciones

Formateo del mensaje de pregunta

El mensaje de pregunta se debe formatear de una de estas dos formas:

- `<cmd:arg1>`
- `<cmd:arg1,arg2>`

El papel de cada una de las partes del mensaje de pregunta es el siguiente:

- `<>` son los delimitadores de mensaje, y se emplean para marcar el inicio y el final del mensaje
- `:` (los dos puntos) son el separador del comando y delimitan el fin de `cmd`
- `,` (la coma) separa los dos argumentos (de existir ambos)
- `cmd` es el comando.
- `arg1` es el primer argumento
- `arg2` es el segundo argumento (opcional)

Formateo del mensaje de respuesta

El formateo del mensaje de respuesta es uno de los siguientes:

- `<ok>`
- `<ok:txt>`
- `<ERROR:txt>`

La función de los indicadores es similar:

- `<>` son los delimitadores de mensaje, y se emplean para marcar el inicio y el final del mensaje
- `:` (los dos puntos) son el separador del texto, si existe
- `ok` indica que el mensaje se ha recibido y procesado correctamente.
- `ERROR` indica que el mensaje no se ha procesado correctamente.
- `txt` es el texto informativo (opcional) y puede servir o para proporcionar un valor o para funciones de debugging.

Posibles valores de los argumentos

El valor del comando `cmd` puede ser o `set` o `get`. En el primer caso se emplea para proporcionar un valor a una variable del microcontrolador, mientras que en el segundo caso se emplea para obtener un valor de una variable.

La tabla 5.1 muestra los posibles valores que puede adoptar `arg1` cuando se emplea el comando `set`.

Valor de `arg1`	Descripción
`mode`	Fija el modo del robot, véase la tabla 5.3
`servo_p`	Fija el ángulo del servo de la posición `p` al valor indicado en `arg2`
`vel`	Fija el valor de $\dot{q}_x$ al valor indicado en `arg2`
`rot`	Fija el valor de la velocidad diferencial al valor indicado en `arg2`

Cuadro 5.1: Valores del primer argumento

Los valores `servo`, `vel` y `rot` sólo están disponibles en el modo de control, y la velocidad diferencial es un valor que se añade a $\dot{q}_x$ en las ruedas del lado derecho y se resta a las del lado izquierdo, de modo que se emplea para girar. `p` puede adoptar los valores `fr`, `fl`, `br` y `bl`, donde `f` indica frontal, `b` indica trasero, `r` indica derecho y `l`, izquierdo.

La tabla 5.2 muestra los posibles valores que puede adoptar `arg1` cuando se emplea el comando `get`.

Valor de `arg1`	Descripción
`mode`	Obtiene el modo del robot, véase la tabla 5.3
`servo_p`	Obtiene el ángulo del servo en la posición `p`
`vel`	Obtiene el valor de $\dot{q}_x$
`rot`	Obtiene el valor de la velocidad diferencial
`d_r`	Obtiene el valor de la distancia en el sensor derecho
`d_l`	Obtiene el valor de la distancia en el sensor izquierdo
`a_x`	Obtiene el valor de la aceleración en el eje x
`a_y`	Obtiene el valor de la aceleración en el eje y
`a_z`	Obtiene el valor de la aceleración en el eje z
`tilt`	Obtiene el valor de la orientación del robot

Cuadro 5.2: Valores del primer argumento

La tabla 5.3 enumera los modos de operación del robot

Modo	Descripción
`0`	Stop: se detiene todo movimiento y para el robot
`1`	Control: el robot es operable desde el PC
`2`	Seguimiento: sigue un objeto o busca la pared más cercana
`4`	Ascenso: busca escalar un escalón
`5`	Descenso: busca descender un escalón

Cuadro 5.3: Modos

5.2. Software del microcontrolador

5.2.1. Programación en Arduino

La plataforma Arduino se puede programar mediante C++. La forma habitual es emplear las siguientes herramientas:

- El IDE de Arduino, que incluye compilador, linker, editor de texto y programador para cargar el programa en la placa.
- Las librerías de Arduino, que proporcionan una capa de abstracción que nos permite acceder a las funciones del microprocesador de una forma sencilla

Ambas herramientas son prescindibles (ya que se puede programar el microcontrolador empleando las herramientas de compilación, generación del archivo hexadecimal y programación por separado y se pueden emplear las funciones del microprocesador "manualmente" asignando las flags correspondientes para cada tarea), pero en este proyecto se han empleado, ya que simplifican considerablemente la tarea de programar para esta plataforma.

La figura 5.2 muestra el flujo de trabajo de un programa desarrollado mediante el IDE de Arduino.

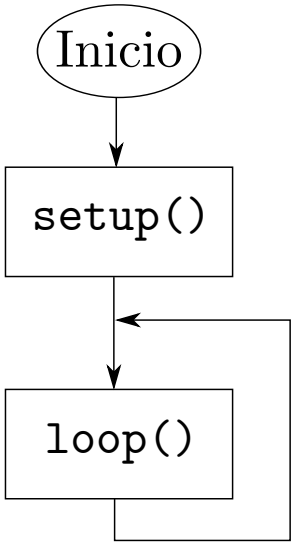

Figura 5.2: Programa en Arduino

El programa debe de constar de dos funciones principales, además de todas las definidas por el usuario:

- La función `void setup()` se ejecuta en primer lugar al arrancar el programa.
- La función `void loop()` se ejecuta a continuación en bucle

5.2.2. Función setup()

En esta función se lleva a cabo la inicialización de las variables. En el software de este proyecto, se encarga de las siguientes funciones:

- Inicializar el puerto serie.
- Inicializar los pines de entrada y salida
- Inicializar los objetos de tipo `SensorIR` (véase la sección **??**)
- Inicializar los objetos de tipo `ServoWithOffset` (véase la sección **??**)
- Inicializar los objetos de tipo `Engine` (véase la sección **??**)
- Inicializar las variables de estado del lector del puerto serie.

5.2.3. Función loop()

Esta función lleva a cabo tres tareas:

- Lectura de los sensores y cálculo de la orientación
- Llamada al método `run()` del objeto de tipo `Model` (véase la sección **??**)
- Tareas de comunicación con el PC

Dado que la señal de los servos es una señal de 50 Hz de frecuencia (lo que limita el periodo de discretización por debajo), se ha decidido leer las entradas y calcular los valores a escribir en las salidas una vez cada 20 ms. Una vez se ha llevado a cabo esa tarea, el programa dedica el resto del ciclo a tareas de comunicación.

La implementación se ha llevado a cabo mediante una variable `next_t`, que almacena (en número de microsegundos desde el arranque del programa) el próximo punto temporal en que se recalculará el modelo cinemático. En ese momento, se incrementa `next_t` en 20 ms (20.000 μs). Se muestra un esquema de este proceso en la figura 5.3

5.2.4. Clase DTrack

En el programa se define una clase `DTrack` que implementa el modelo cinemático del robot e incluye las variables de estado para separar los movimientos de los comportamientos autónomos. La interfaz de la clase está compuesta por los métodos de la tabla 5.4

Función	Descripción
`set_mode()`	Selecciona el comportamiento del robot
`run()`	Actualiza el modelo cinemático y escribe los nuevos valores en las salidas
`set_servo(pos, val)`	(Modo de control) Coloca el servo en la posición `pos` en el ángulo `val`
`set_vel(v)`	(Modo de control) Asigna a $\dot{q}_x$ el valor `v`
`def_dv(dv)`	(Modo de control) Iguala la velocidad diferencial a `dv`
`get_servo(pos)`	Devuelve el ángulo del servo en la posición `pos`
`get_vel()`	Devuelve el valor de $\dot{q}_x$
`get_dv()`	Devuelve el valor de la velocidad diferencial

Cuadro 5.4: Métodos de la clase DTrack

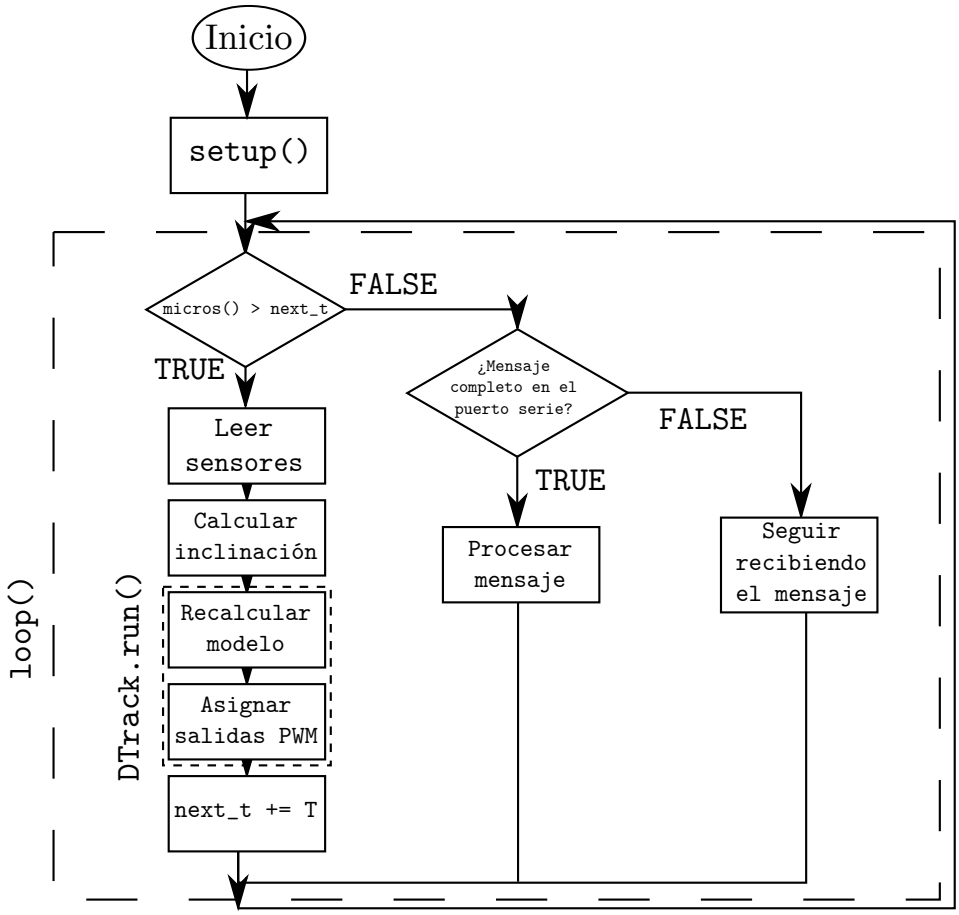

Figura 5.3: Diagrama de la función loop()

5.2.5. Otras clases

SensorIR

La clase `SensorIR` es una capa de abstracción para leer el valor de los sensores. Su interfaz se presenta en la tabla 5.5

Función	Descripción
`set_pin(p)`	Asigna al sensor el pin `p`
`get_distance()`	Calcula la distancia según la fórmula de la sección 4.3.1

Cuadro 5.5: Métodos de la clase SensorIR

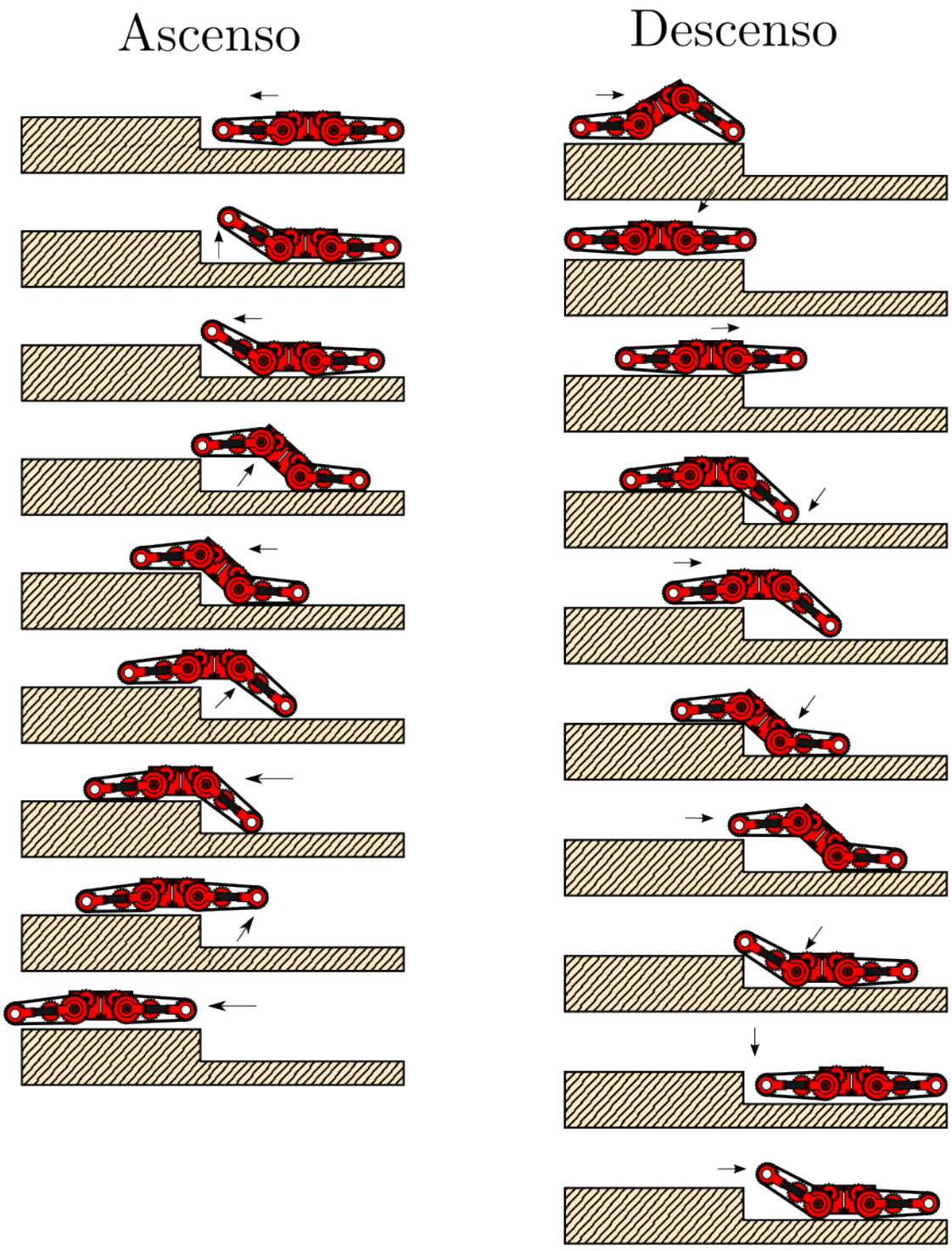

Figura 5.4: Pasos de las maniobras autónomas

ServoWithOffset

La clase `SensorWithOffset` encapsula la clase `Servo` de la librería de Arduino, pero permite especificar un offset para el valor del ángulo. Su interfaz se presenta en la tabla 5.6

Función	Descripción
`attach(p)`	Asigna el pin `p` al pin del servo
`write(angle)`	Coloca el servo en la posición `angle`
`set_offset(offset)`	Asigna el valor del offset
`reverse()`	Invierte el movimiento del servo

Cuadro 5.6: Métodos de la clase ServoWithOffset

Engine

La clase `Engine` también encapsula la clase `Servo` de la librería de Arduino, pero el objetivo es controlar un servo modificado para rotación continua y no un servo normal. Su interfaz se presenta en la tabla 5.7

Función	Descripción
`attach(p)`	Asigna el pin `p` al pin del motor
`set_v(vel)`	Hace girar al servo con velocidad `vel`
`set_dz(dz1, dz2)`	Inicializa la deadzone de `dz1` a `dz2`
`reverse()`	Invierte el movimiento del servo

Cuadro 5.7: Métodos de la clase Engine

5.2.6. Otras funciones

Otras funciones creadas, además de `setup()` y `loop()`, son las siguientes:

- `rads_to_rpm(omega)` convierte de radianes por segundo a revoluciones por minuto.
- `transmit(s)` formatea una cadena de caracteres y la envía por el puerto serie.
- `compute_tilt()` lee los valores del acelerómetro y calcula la orientación del robot.
- `receive_cycle()` comprueba si hay caracteres de entrada en el puerto serie y, en tal caso, procesa el mensaje y almacena los valores del comando y los argumentos en variables globales
- `communicate()` procesa el comando y toma acciones en consecuencia.

5.3. Software del PC

El software de control del PC es sencillo. Consta de una capa de abstracción para acceder al robot en la forma de la clase `DTrackSerial` que implementa el protocolo diseñado y de un bucle que gestiona una interfaz basada en texto. La tabla 5.8 enumera los métodos de la clase mencionada:

Función	Descripción
`connect(port)`	Inicia la conexión en el puerto `port`
`get_data(key)`	Obtiene el dato `key`
`set_data(key, value)`	Asigna el valor `value` al dato `key`

Cuadro 5.8: Métodos de la clase DTrackSerial

El programa incluye una interfaz textual, que se muestra en la figura 5.4:

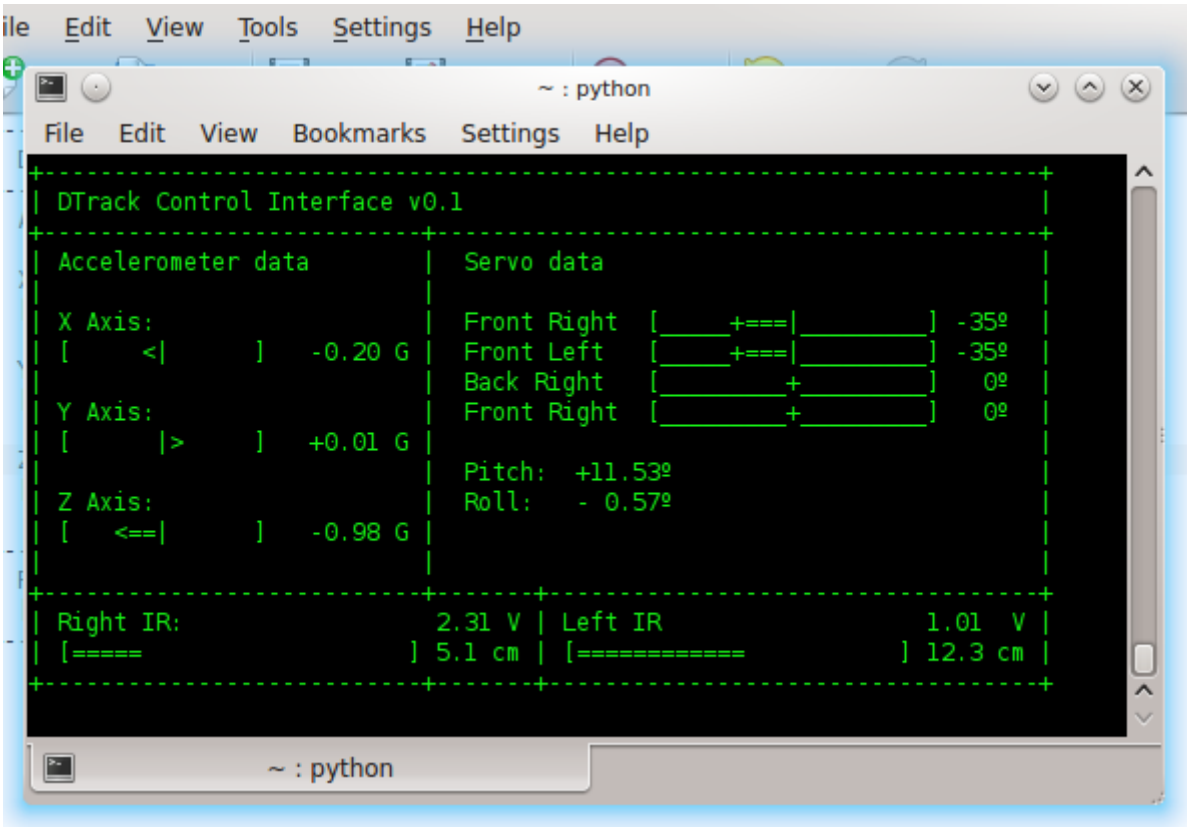

Figura 5.5: Interfaz de control

El robot se controla (si el modo lo permite) mediante las teclas señaladas en la tabla 5.9

Tecla	Función
`Q`	Levanta el track delantero izquierdo
`W`	Levanta el track delantero derecho
`E`	Levanta el track trasero izquierdo
`R`	Levanta el track trasero derecho
`A`	Hace descender el track delantero izquierdo
`S`	Hace descender el track delantero derecho
`D`	Hace descender el track trasero izquierdo
`F`	Hace descender el track trasero derecho
`I`	Aumenta la velocidad del robot
`K`	Disminuye la velocidad del robot
`L`	Aumenta la velocidad diferencial del robot
`J`	Disminuye la velocidad diferencial del robot
`0`	Para el robot
`1`	Coloca el robot en modo de control
`2`	Coloca el robot en modo de seguimiento
`3`	Coloca el robot en modo de ascenso
`4`	Coloca el robot en modo de descenso

Cuadro 5.9: Teclas de control

Capítulo 6

Montaje

El proceso de montaje de un vehículo puede adoptar formas muy diferentes dependiendo del ámbito, las características y el tipo de proyecto: un vehículo comercial, un prototipo de investigación o un vehiculo de competición pueden ser fabricados mediante procesos de montaje con diferentes características de automatización, complejidad o maquinaria. El montaje de un robot imprimible como el que describe este proyecto es un proceso sencillo comparado con los tipos de proyecto mencionados, y se realiza de forma manual; no obstante, es necesario ejecutar las operaciones de montaje de forma correcta y en el orden correspondiente.

Este capítulo se divide en dos secciones:

- La sección 6.1 enumera las piezas, materiales y herramientas necesarias.
- La sección 6.2 describe el proceso de montaje

6.1. Materiales

6.1.1. Piezas imprimibles

La tabla 6.1 enumera las piezas necesarias y la cantidad requerida.

#	Nombre de la pieza	Cantidad necesaria
1	Pieza exterior del track	8
2a	Pieza interior del track larga	4
2b	Pieza interior del track corta	4
3a	Pieza de unión del track	6
3b	Pieza de unión con soporte de sensor	2
4	Engranaje de la rueda	4
5	Engranaje del track	4
6	Rueda pequeña	4
7	Cuerpo de la rueda grande	4
8	Tapa de la rueda grande	8
9	Asiento de la rueda grande	4
10	Rodamiento cilíndrico	44
11	Pieza del eje	4
12	Tapón del eje	4
13	Pieza de la base	2

Cuadro 6.1: Piezas imprimibles

6.1.2. Componentes adicionales

Los elementos no imprimibles necesarios se enumeran en la tabla 6.2.

Objeto	Cantidad necesaria
Tornillo M4 de 12 mm de longitud	32
Tornillo M4 de 20 mm de longitud	24
Tornillo M4 de 30 mm de longitud	16
Tornillo M4 de 80 mm de longitud	2
Tornillo M6 de 80 mm de longitud	4
Tornillo M3 de 10 mm de longitud	4
Tornillo M2.5 de 12 Tuerca M4	78
Tuerca M6	4
Tuerca M3	4
Cable con conector JST de 3 pines	2

Cuadro 6.2: Piezas no imprimibles

6.1.3. Herramientas

Durante el montaje del robot son útiles las siguientes herramientas:

- Un destornillador plano con una cabeza de unos 3 mm de longitud. Mediante un destornillador de estas dimensiones o similares podemos atornillar la práctica totalidad de los tornillos que unen las piezas del robot, con la excepción de los tornillos de rosca métrica M6.
- Un destornillador plano con una cabeza de 6 mm. Este destornillador o uno con una cabeza superior es de gran ayuda para atornillar los tornillos M6.
- Un destornillador de estrella con una cabeza pequeña, de unos 2 mm de diámetro, es necesario para atornillar la corona al servo.
- Unos alicates también ayudan durante el proceso de atornillado, sujetando la tuerca. Adicionalmente, el alicate es útil para recortar la corona del servo y crear la pieza compuesta por engranaje y una corona del servo.
- La acetona se emplea para pegar la pieza del eje con el tapón y para pegar las piezas interiores del track con las exteriores según el procedimiento explicado en la sección 2.3.4
- También se empleará un pegamento para plásticos de dos componentes, también para unir la corona recortada del servo con el engranaje del track o de la rueda, como se menciona en la sección 2.3.5
- Una herramienta rotativa es útil para ampliar los agujeros para tornillos en los que existan rebabas o el hueco no sea suficientemente grande por algún motivo (sección 2.3.2)
- Un cúter permite recortar aristas puntiagudas (por ejemplo, por causa de rebabas) y ajustar la dimensión de la corona del servo para permitir su encaje en los engranajes

6.2. Proceso de montaje

El proceso de montaje se puede dividir, a grandes rasgos, en seis partes:

- Montaje de la parte interior del track (subsección 6.2.1)
- Montaje de la parte exterior del track (subsección 6.2.2)
- Montaje de la rueda grande (subsección 6.2.3)
- Montaje de la pieza del eje (subsección 6.2.4)
- Montaje del conjunto track-articulación (subsección 6.2.5)
- Unión del conjunto track-articulación con las bases (subsección 6.2.6)

En la figura 6.1 se ilustran las relaciones entre las piezas, los elementos de unión y los motores y sensores.

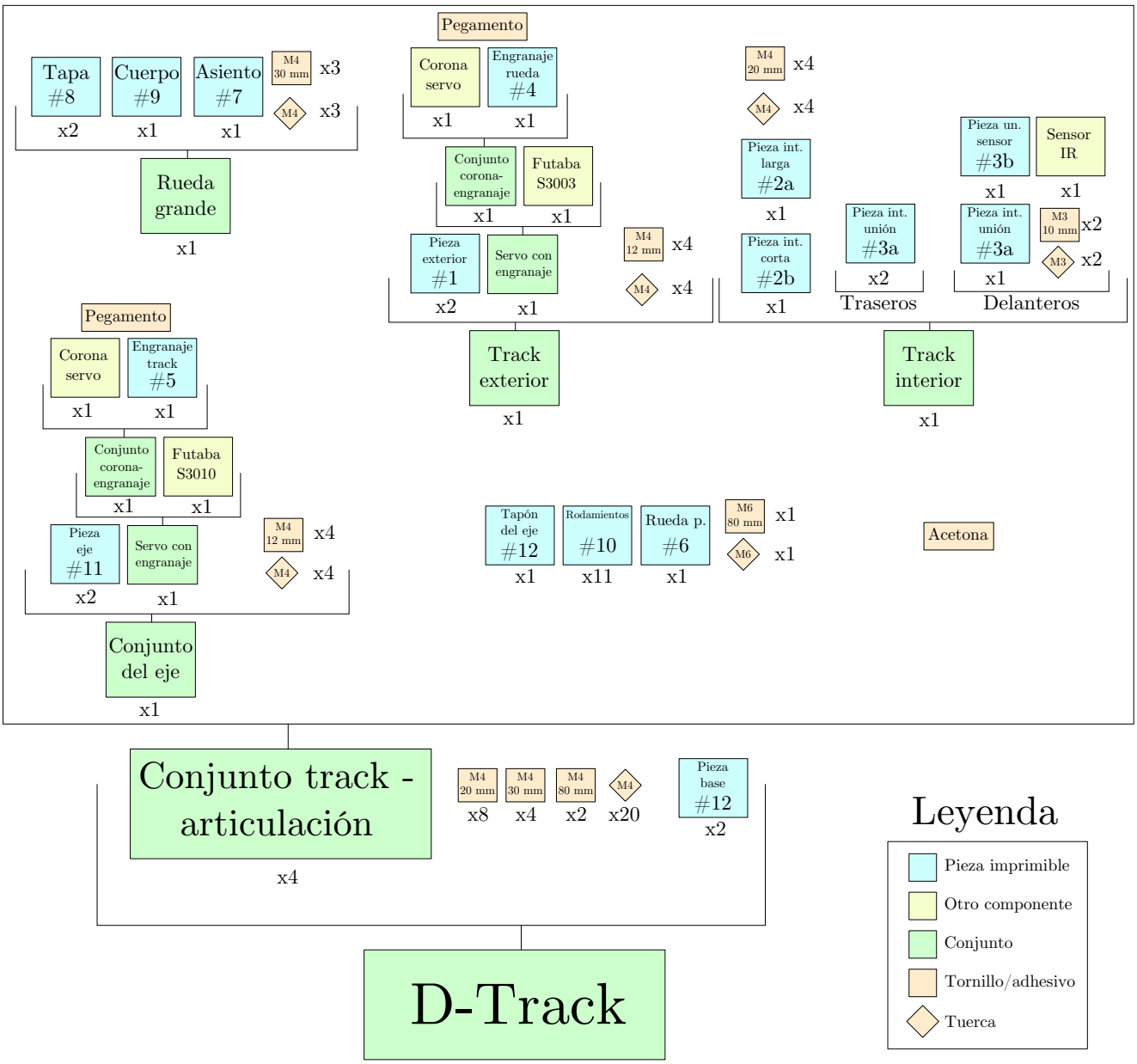

Figura 6.1: Diagrama del montaje

6.2.1. Montaje de la parte interior del track

Hay dos tipos de tracks: los delanteros y los traseros. Ambos se montan de forma muy similar, exceptuando el hecho de que los delanteros deben montar la pieza del sensor.

Montaje del sensor

En primer lugar, debemos conectar un cable JST al sensor Sharp GP2D120XJ00F:

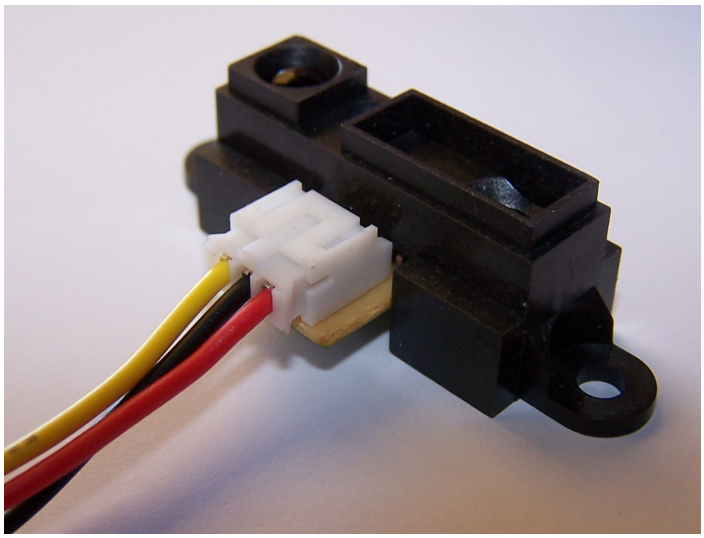

Figura 6.2: Sensor IR con cable

A continuación, debemos atornillar el sensor al soporte (pieza #3b) mediante dos tornillos M3 de 10 mm de longitud y sus correspondientes tuercas.

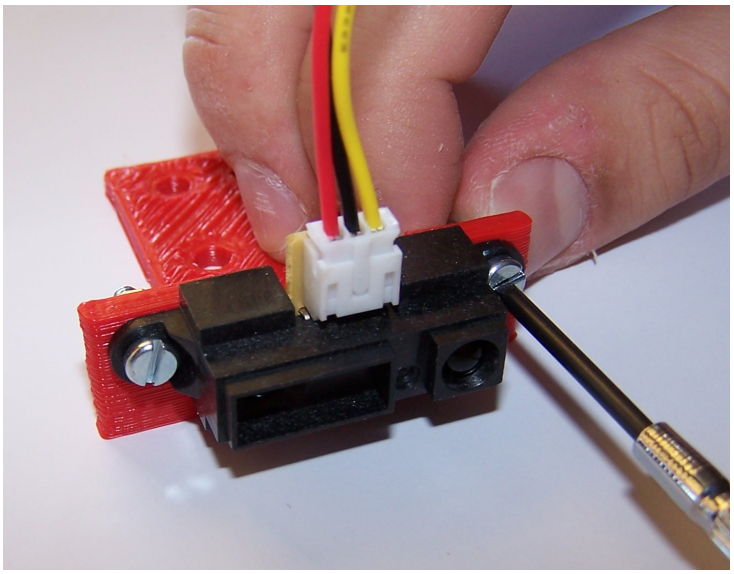

Figura 6.3: Sensor IR en su soporte

Resto del montaje

Para la parte interior del track trasero uniremos la pieza interior corta (2b) y la pieza interior larga (2a) con dos piezas de unión (3a) mediante 4 tornillos M4 de 20 mm.

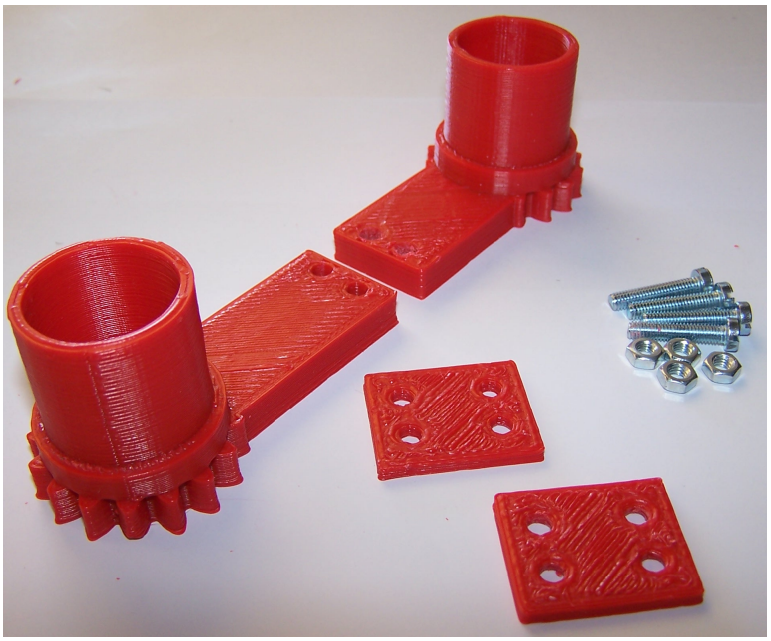

Figura 6.4: Piezas del track interior

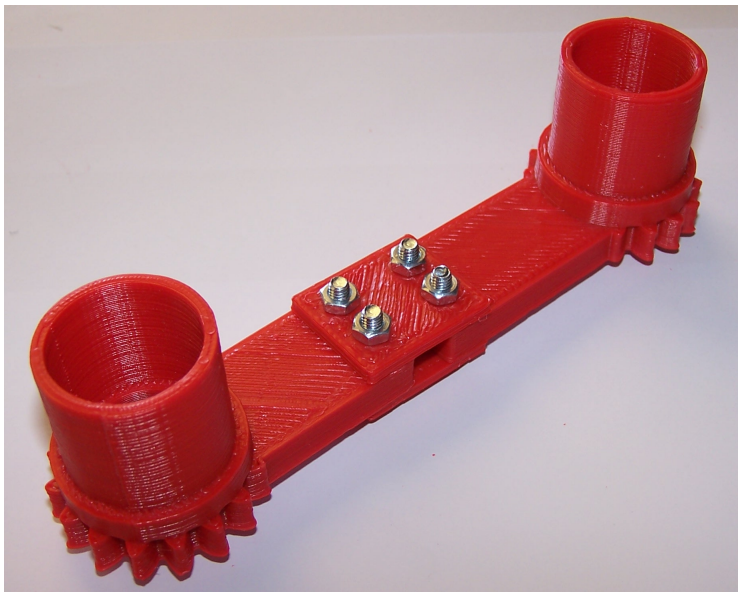

Figura 6.5: Parte interior del track trasero, montada

En el caso de que sea un track delantero, sustituiremos una de las piezas de unión (3a) por una pieza de unión con soporte (3b) con el sensor montado. El montaje se puede observar en la figura 6.6:

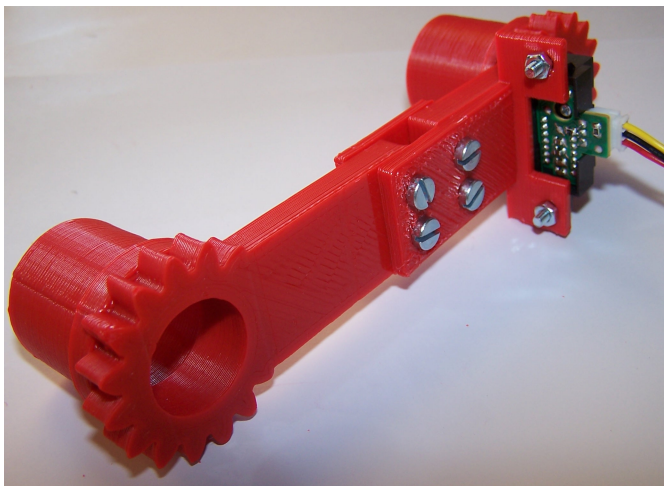

Figura 6.6: Parte interior del track delantero, montada

6.2.2. Montaje de la parte exterior del track

El primer paso para montar la parte exterior del track es desatornillar una corona de servo y cortar un lado con unos alicates de corte:

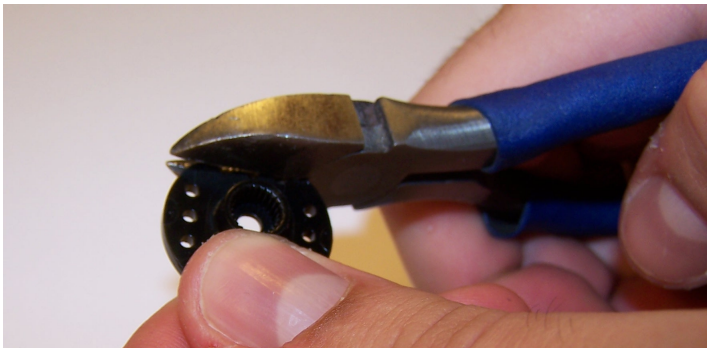

Figura 6.7: Corte de la corona del servo

A continuación, se aplicará la imprimación del pegamento a la cara exterior de la corona y al hueco en el engranaje de la rueda (pieza 4); tras esperar unos segundos para que se seque, se aplicará el otro componente del pegamento de plásticos al hueco y se encajarán las piezas a presión.

Este conjunto se atornillará a un servo Futaba S3003 modificado mediante su tornillo correspondiente, como se ilustra en la figura 6.9

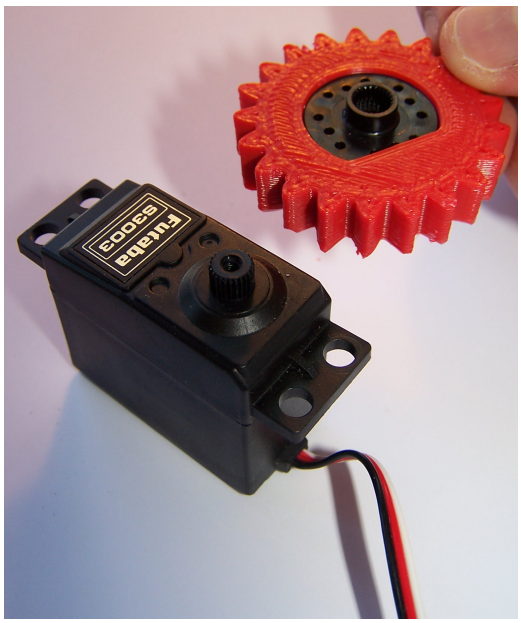

Figura 6.8: Conjunto engranaje-corona

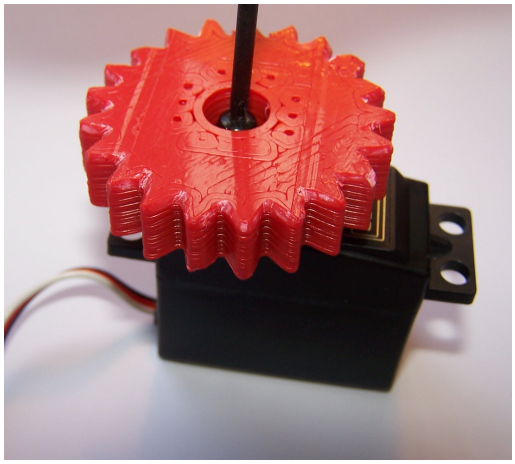

Figura 6.9: Atornillamiento del servo con el engranaje de la rueda

Finalmente, completaremos el montaje de la parte exterior del track uniendo el servo a las piezas del track exteriores (pieza #1) mediante tornillos M4 de 12mm de longitud y sus tuercas correspondientes.

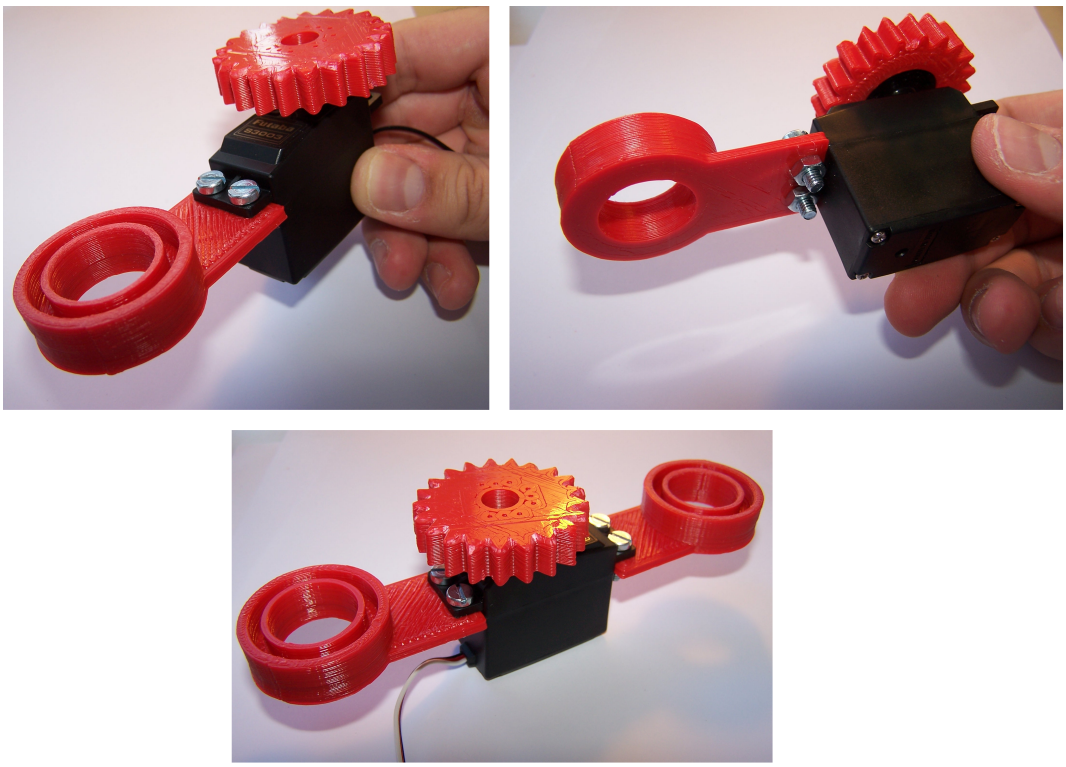

Figura 6.10: Montaje de la parte exterior del track

6.2.3. Montaje de la rueda grande

La rueda grande está compuesta por dos tapas (pieza #8), un cuerpo de la rueda (pieza #9) y un asiento o rodamiento cilíndrico (pieza #10).

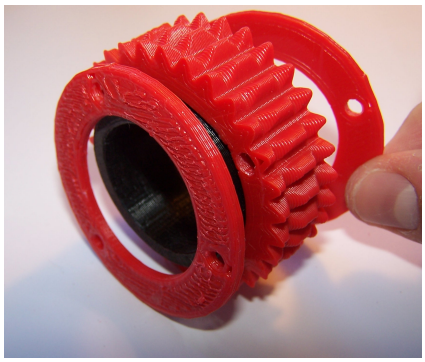

Figura 6.11: Piezas de la rueda grande

El montaje consiste en unir las piezas tal y como se indica en las figuras 6.11 y 6.12 mediante tornillos M4 de 30 mm de longitud y tuercas M4.

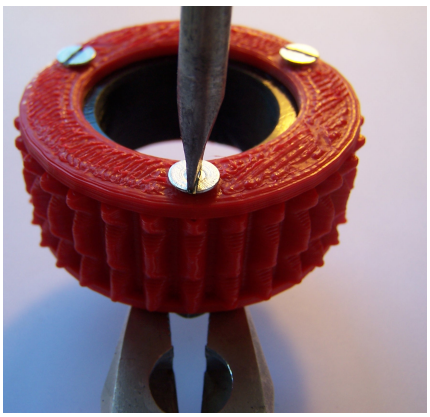

Figura 6.12: Montaje de la rueda grande

6.2.4. Montaje de la pieza del eje

Para montar el eje, debemos seguir el procedimiento explicado en la sección 6.2.2 para crear un conjunto engranaje-corona empleando un engranaje de track en lugar de un engranaje de rueda.

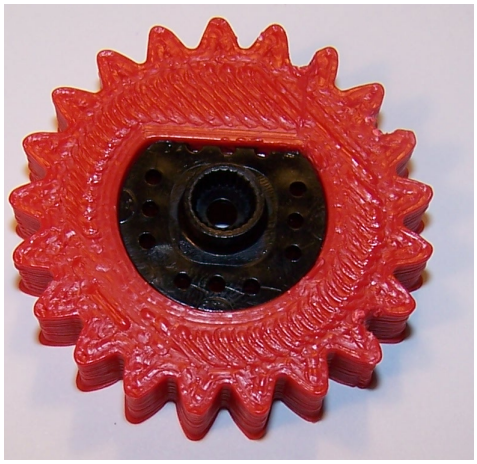

Figura 6.13: Engranaje del track

A continuación, montaremos un servo Futaba S3010 sin corona en la pieza del eje (#11) mediante tornillos M4 de 12 mm y tuercas M4 según la imagen 6.14

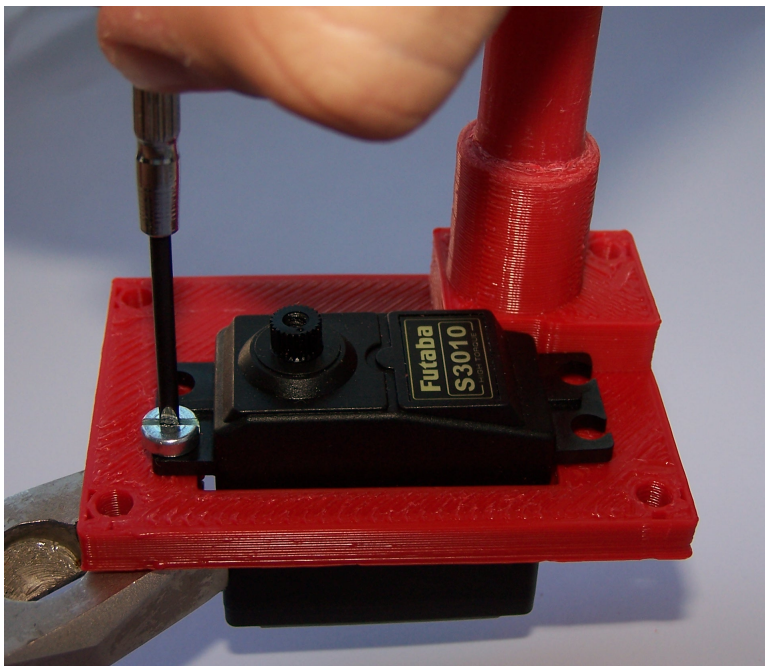

Figura 6.14: Montaje del S3010 en la pieza del eje

Y atornillamos la corona:

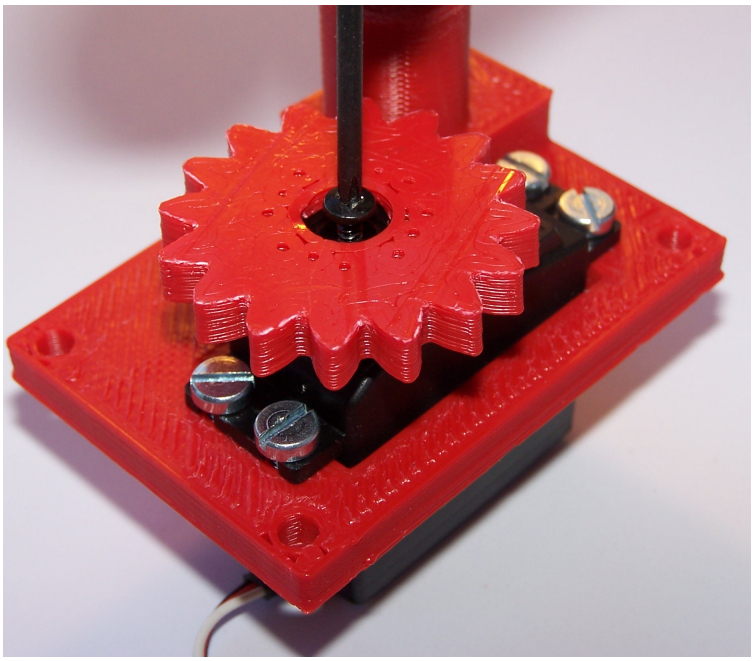

Figura 6.15: Montaje de la corona del S3010

Finalmente, giraremos la corona de modo que el servo quede en la posición media

6.2.5. Montaje del conjunto track-articulación

Con una pieza del eje con el S3010 montado según la sección 6.2.4, colocaremos la parte interior del track en el eje de forma que quede aproximadamente paralela al S3010. En este momento no debe quedar fija, ya que no se han instalado los rodamientos.

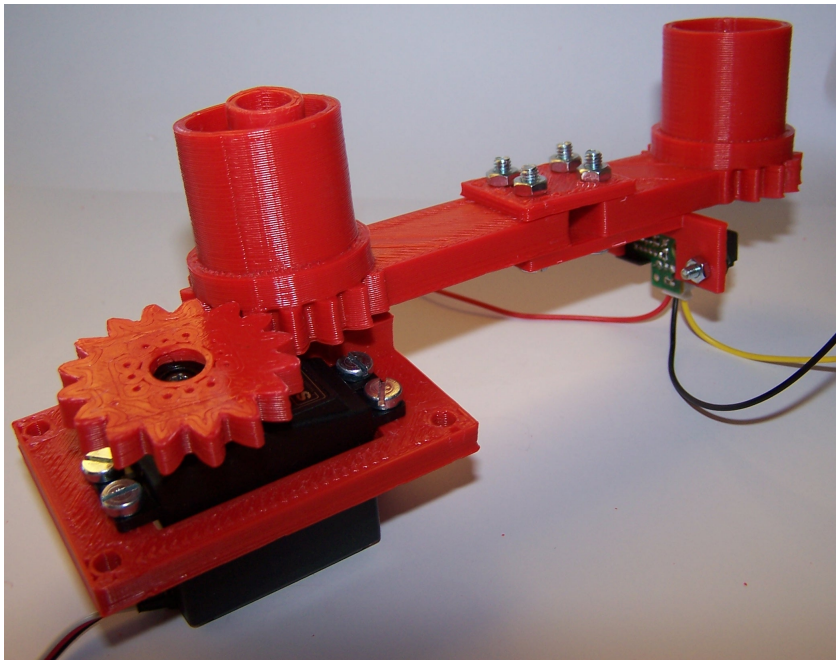

Figura 6.16: Montaje de la articulación, primer paso

A continuación, se colocarán las ruedas en sus respectivos lugares. La rueda grande debe quedar colocada con las tuercas hacia el exterior.

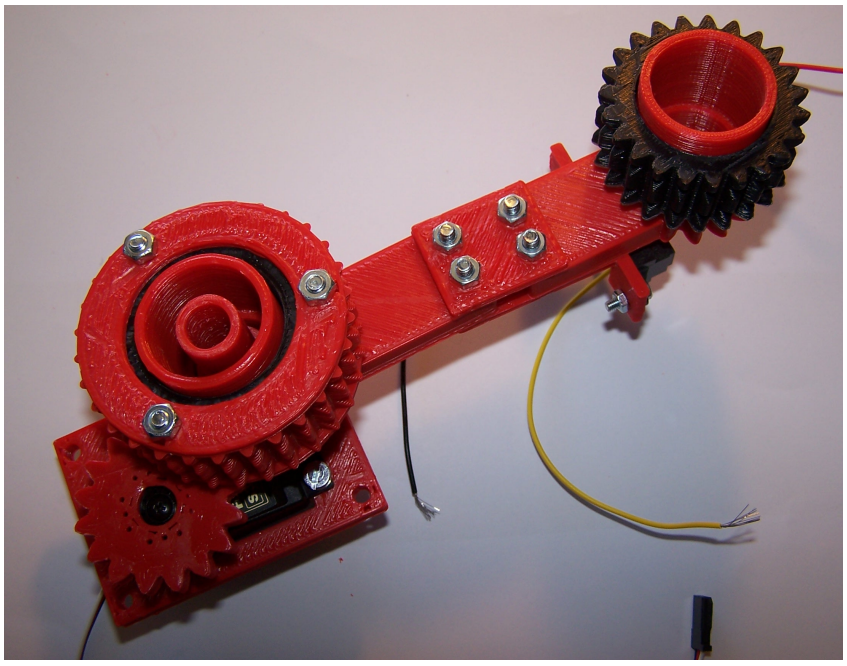

Figura 6.17: Montaje de la articulación, segundo paso

Posteriormente, se introducirá un tornillo M6 de 80 mm por el agujero del eje.

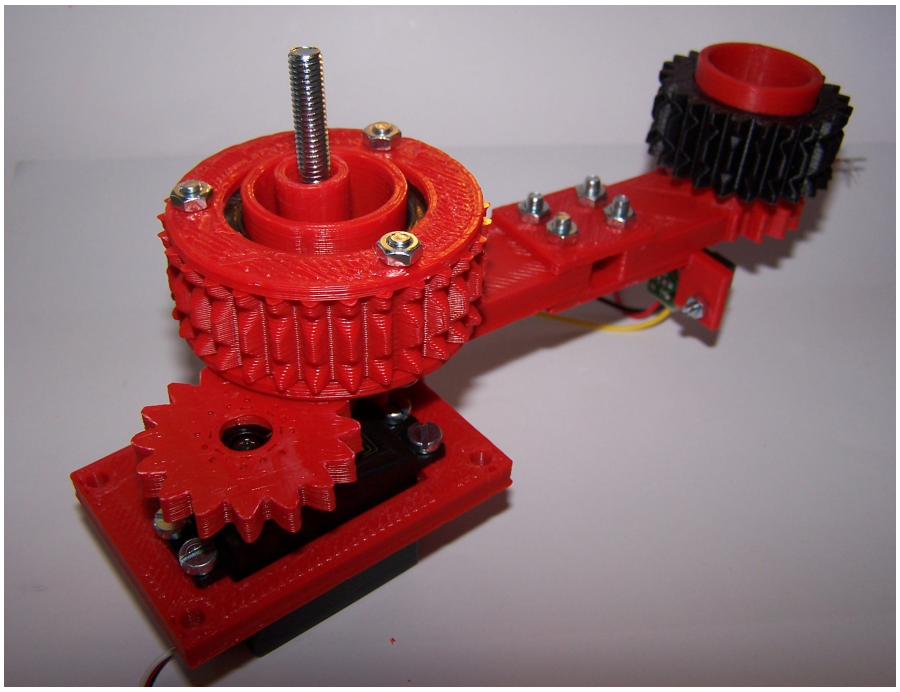

Figura 6.18: Montaje de la articulación, tercer paso

Después de introducir el tornillo, colocaremos 11 rodamientos cilíndricos (pieza #10) entre el eje y la pieza interior del track.

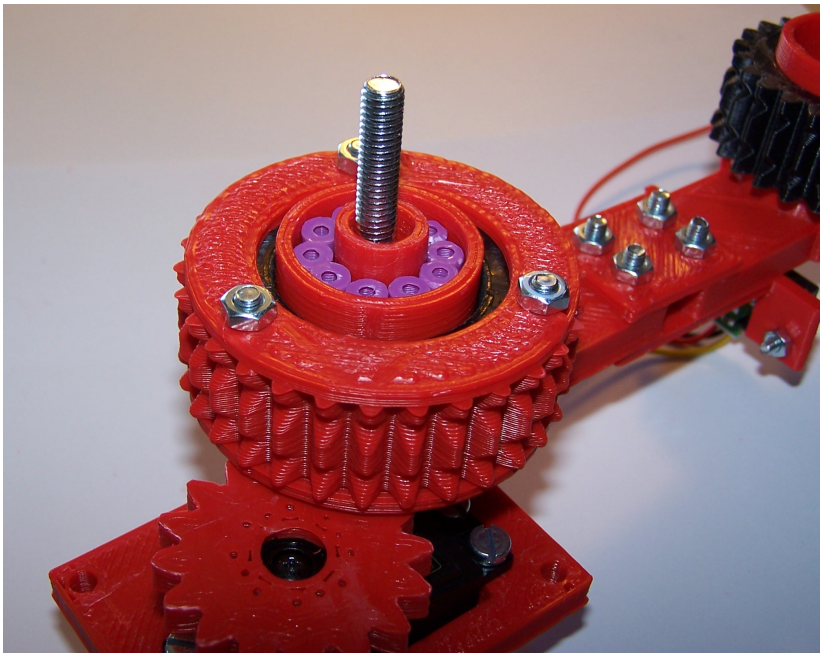

Figura 6.19: Montaje de la articulación, cuarto paso

Tras colocar los rodamientos, colocaremos el tapón (pieza #12) en el eje y lo fijaremos mediante la tuerca M6.

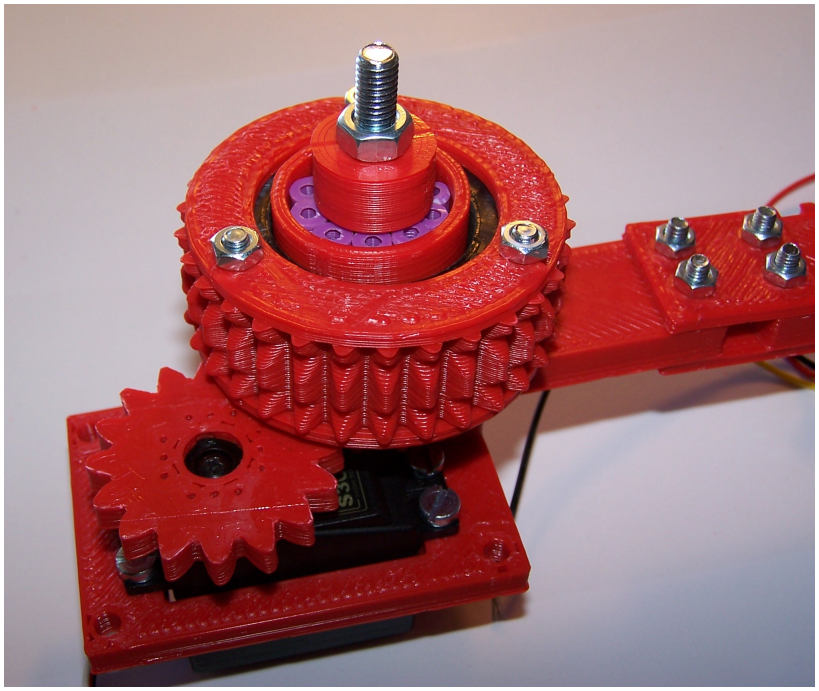

Figura 6.20: Montaje de la articulación, quinto paso

Finalmente, aplicaremos acetona en el borde de la pieza interior del track y encajaremos la pieza exterior, con el servo engranando en la rueda grande.

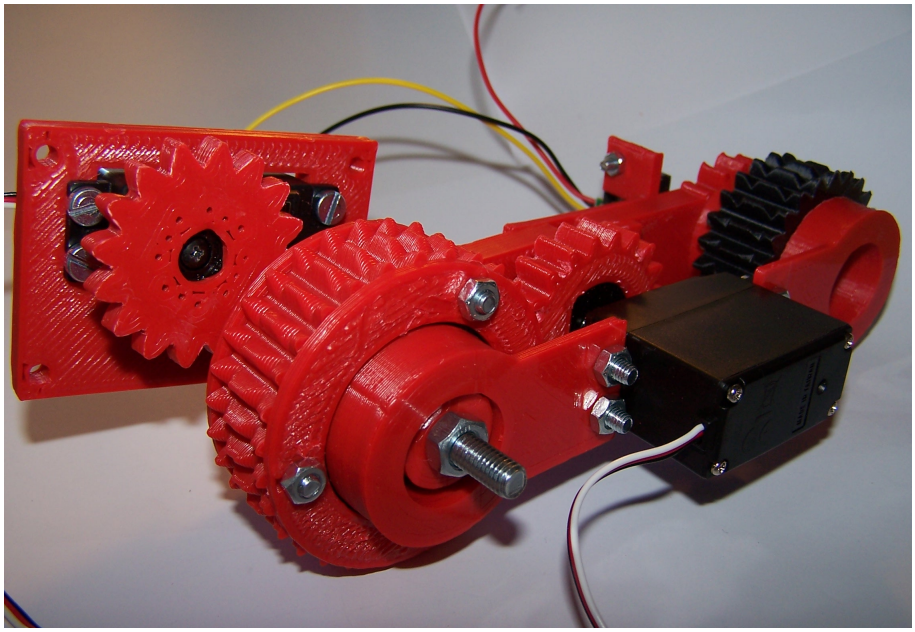

Figura 6.21: Montaje de la articulación, sexto paso

6.2.6. Unión del conjunto track-articulación con las bases

En primer lugar, atornillaremos el conjunto track-articulación a la base inferior (pieza #12) mediante un tornillo M4 de 30 cm (extremo) y un tornillo M4 de 20cm (centro) y sus respectivas tuercas.

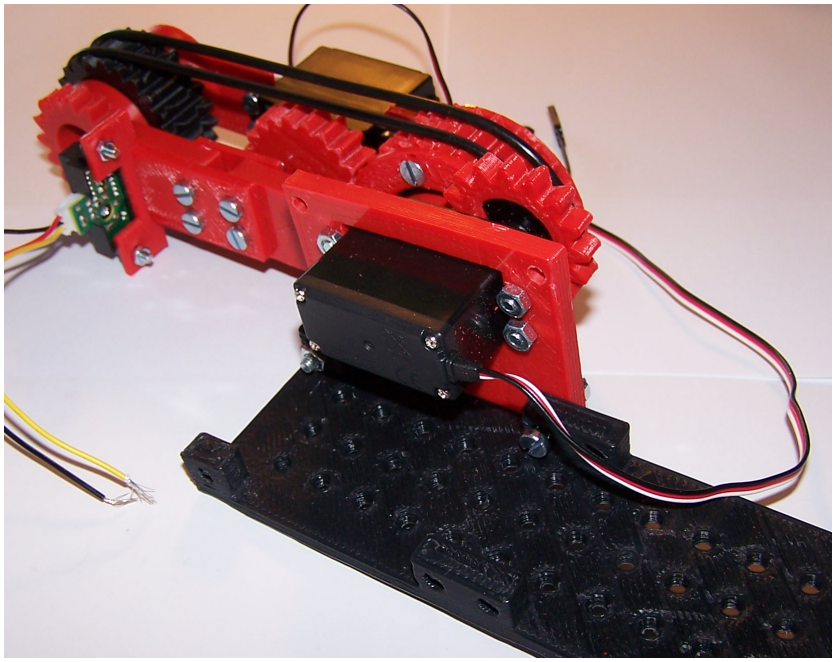

Figura 6.22: Montaje de la base inferior

Repetiremos este paso con todos los tracks. Por último, fijaremos la base superior con tornillos M4 de 20 mm en los extremos y dos tornillos M4 de 80 mm en las uniones del centro. A estos tornillos se les colocarán dos tuercas en la cabeza para evitar que sobresalgan demasiado por el lado opuesto.

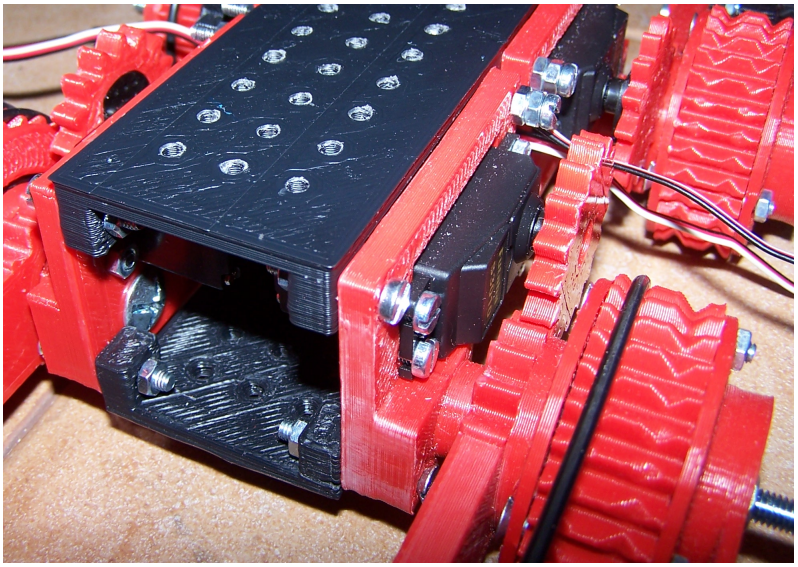

Figura 6.23: Montaje de la base superior

Finalmente, el robot queda completado colocando las correas:

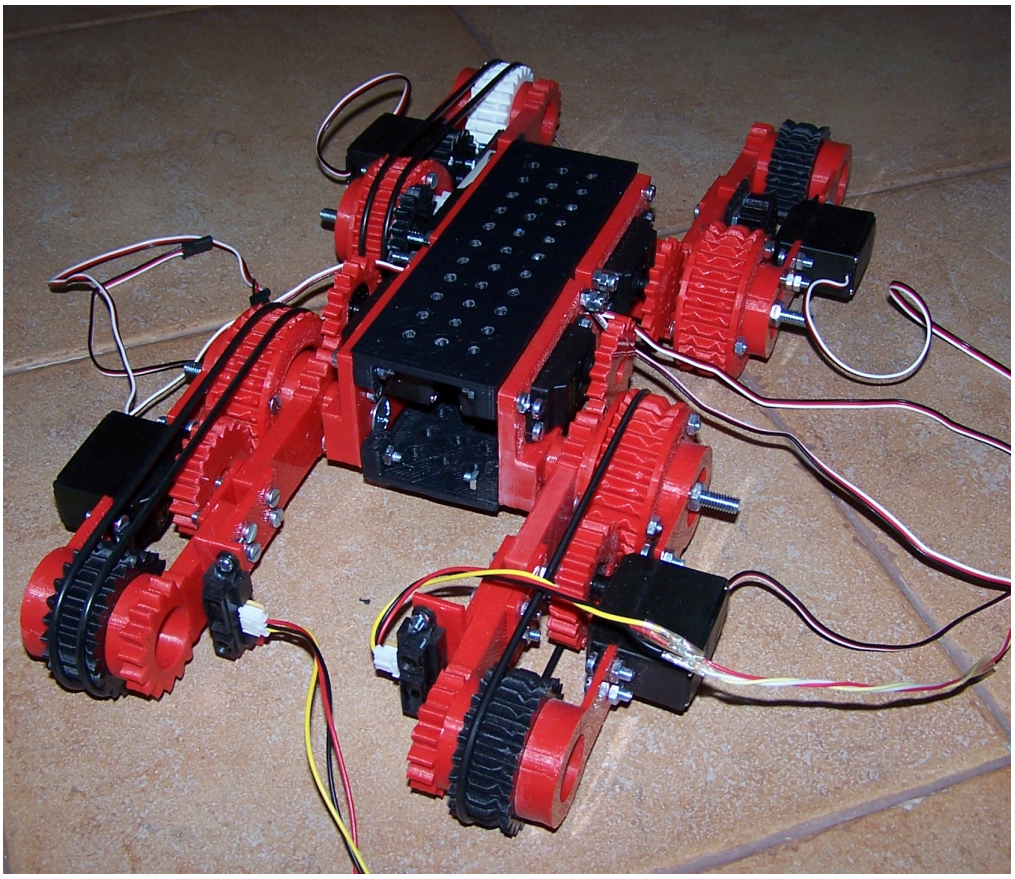

Figura 6.24: Colocando las correas

Capítulo 7

Conclusiones y trabajo futuro

El presente capítulo recoge las conclusiones del autor del proyecto y su evaluación del grado de cumplimiento de los objetivos del proyecto. Adicionalmente, en este capítulo se plantean líneas de trabajo futuras en los aspectos mecánicos, electrónicos e informáticos del robot, de forma que amplíen la funcionalidad del robot o mejoren la funcionalidad ya existente.

El capítulo está formado por 5 secciones:

- La sección 7.1 expone las conclusiones del autor acerca del proyecto.
- La sección 7.2 explica lo que ha supuesto la elaboración del proyecto para el autor
- La sección 7.3 enumera una serie de propuestas de mejora de la mecánica del vehículo
- La sección 7.4 presenta líneas de trabajo relativas a la electrónica.
- La sección 7.5 propone varias mejoras relativas al software del robot.

7.1. Conclusiones generales

Al término del proyecto, las conclusiones son muy positivas, ya que los objetivos del proyecto se han cumplido satisfactoriamente:

- Se ha desarrollado desde cero un robot con amplias posibilidades motrices, puesto que es capaz de operar en relieves complicados en los que robots basados en ruedas o sistemas similares de tracción tendrían problemas; adicionalmente, el sistema basado en tracks preserva gran parte de las ventajas de los robots basados en ruedas, como la eficiencia de este método de tracción.

- El robot desarrollado arregla los principales problemas del F-Track original: el pandeo de los tracks se ha solucionado creando un eje propio para el track y proporcionándole tracción mediante engranajes, los problemas de tracción de las ruedas se han solucionado sustituyendo la transmisión basada en correas con engranajes, se ha reducido el peso de los tracks y se han solucionado las carencias de potencia que presentaba el F-Track original mediante un diseño más favorable y el empleo de servos más potentes. Adicionalmente, se han introducido varias mejoras: se han instalado sensores, se ha reformado el cuerpo central (de modo que permite la instalación de más equipamiento) y el nuevo diseño, al ser en gran parte desmontable, es más favorable a la reforma, la reparación y la ampliación.

- Se han empleado en gran parte tecnologías libres para el diseño, la fabricación y la electrónica del robot. Se han fabricado las piezas del robot con impresoras 3D del proyecto RepRap, se ha empleado un microcontrolador Arduino y se ha diseñado en una plataforma de desarrollo basada en software libre. Se han diseñado las piezas en un formato paramétrico fácilmente modificable (OpenSCAD) y se ha completado un diseño favorable para la replicación y la ampliación.

- Se ha caracterizado matemáticamente el robot en diversas situaciones de operación y se ha implementado el modelo matemático para realizar un control efectivo del vehículo.

- Se ha desarrollado un programa que demuestra las capacidades del robot y que proporciona una vía natural para ampliar la funcionalidad del vehículo o crear interfaces con otros dispositivos o tecnologías. El software implementado incluye varias maniobras autónomas que se apoyan en el sistema de percepción del robot, sentando la base para un diseño completamente autónomo.

- Se han demostrado las ventajas de la robótica imprimible de bajo coste, entre las que se encuentran la accesibilidad de las tecnologías empleadas, la amplia disponibilidad de herramientas de desarrollo, el rápido ciclo de desarrollo, el reducido coste de materiales, la adaptabilidad y la modificabilidad. Se ha caracterizado la metodología de desarrollo de robots imprimibles basados en las tecnologías utilizadas.

En conclusión, hay abundantes motivos para estar satisfecho de los resultados del presente proyecto.

7.2. Conclusiones personales

Desarrollar el presente trabajo fin de grado ha supuesto una experiencia muy enriquecedora para mí. La creación del D-Track ha sido un proyecto que ha involucrado el empleo de conocimientos de diversas áreas de la ingeniería, la física y las matemáticas; desde la mecánica hasta el desarrollo de software, pasando por la electrónica, la robótica, el álgebra o la estadística. Me ha resultado especialmente gratificante la aplicación de los contenidos estudiados durante estos años para resolver problemas prácticos así como la obtención de resultados tangibles, dándome ánimos para continuar mi formación como ingeniero y eventualmente poner mis conocimientos al servicio de la sociedad.

Durante la elaboración de este proyecto no sólo he aprendido conceptos de ingeniería; este proyecto me ha recordado la utilidad de la organización, la disciplina, la planificación, la paciencia y la perseverancia. Me ha hecho ver la importancia de la metodología de trabajo, me ha mostrado la fuerza de la ilusión y me ha enseñado el valor de la creatividad. Me ha hecho valorar diferentes soluciones alternativas para todo tipo de problemas, y a enfrentarme a los trade-offs que aparecen. Me ha conducido a desempolvar libros de materias que creía olvidadas, y a abrir otros tantos sobre tecnologías que ni siquiera conocía.

Termino este trabajo muy satisfecho por todo lo que ha significado para mí, y con la esperanza de que quizás sirva de inspiración a otros para aventurarse en el fascinante mundo de los robots imprimibles de igual forma que otros me inspiraron a mí a llevar a cabo este proyecto.

7.3. Mejoras mecánicas

A continuación se presentan algunas posibles líneas de trabajo futuras relativas a la mecánica del vehículo:

7.3.1. Sustitución de los tracks traseros por un único track giratorio

La morfología actual del robot no favorece el giro: si bien es posible girar el robot aplicando una velocidad diferencial a las ruedas de un lado respecto a las del otro, es una operación que involucra deslizamiento y por lo tanto es lenta y aparatosa. Una posible mejora sería la sustitución de la parte trasera del vehículo por una articulación giratoria unida a un único track, en una configuración de "triciclo inverso". La figura 7.1 presenta un ejemplo de dicha configuración.

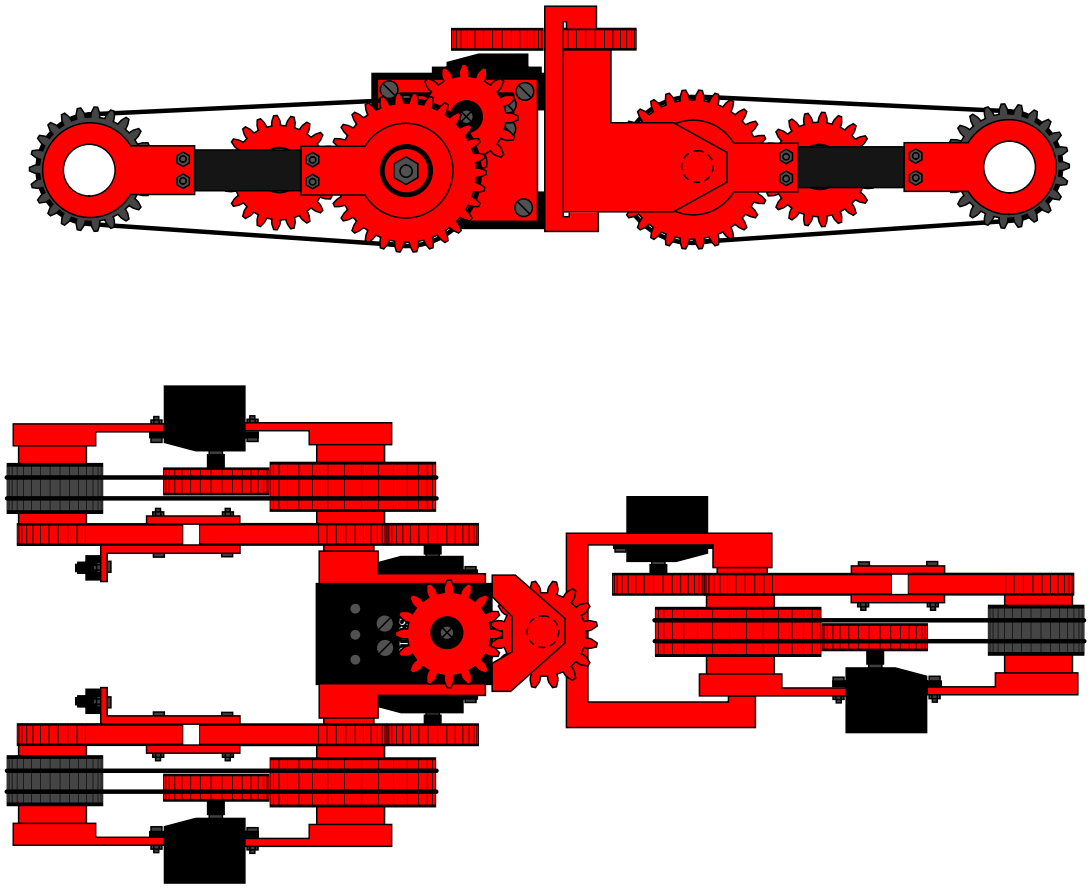

Figura 7.1: Configuración de triciclo

Esta configuración preserva la mayoría de la funcionalidad de la configuración del D-Track y el F-Track original (sólo pierde potencia en la parte trasera al haber sólo un servo levantando más peso), pero gana una capacidad de giro sustancial.

7.3.2. Adición de instrumentos

En la base superior es posible atornillar instrumentos adicionales: por ejemplo, un brazo robótico o una pinza. La adición de estos componentes permitiría al robot no sólo percibir y navegar el terreno, sino interactuar con el entorno y recoger y transportar objetos.

7.3.3. Sustitución de la transmisión entre ruedas mediante correas

Una idea a considerar es llevar más lejos el uso de engranajes en el D-Track y conectar directamente las ruedas mediante engranajes, reemplazando de este modo la transmisión basada en correas. La forma más sencilla de llevar esta idea a cabo sería acortar la longitud del track y engranar el engranaje del motor directamente a ambas ruedas. Dado que la velocidad lineal en la circunferencia de cada engranaje sería idéntica sea cual sea la relación de transmisión entre el engranaje y una rueda y el engranaje y la otra.

7.3.4. Sustitución de las correas por orugas imprimibles

Otro posible sistema que podría reemplazar las correas como método de transmisión entre ruedas sería una oruga imprimible. La figura 7.2 muestra el Orugator o Caterpillator, un robot diseñado por Olalla Bravo y Daniel Gómez que se caracteriza por emplear orugas imprimibles para moverse.

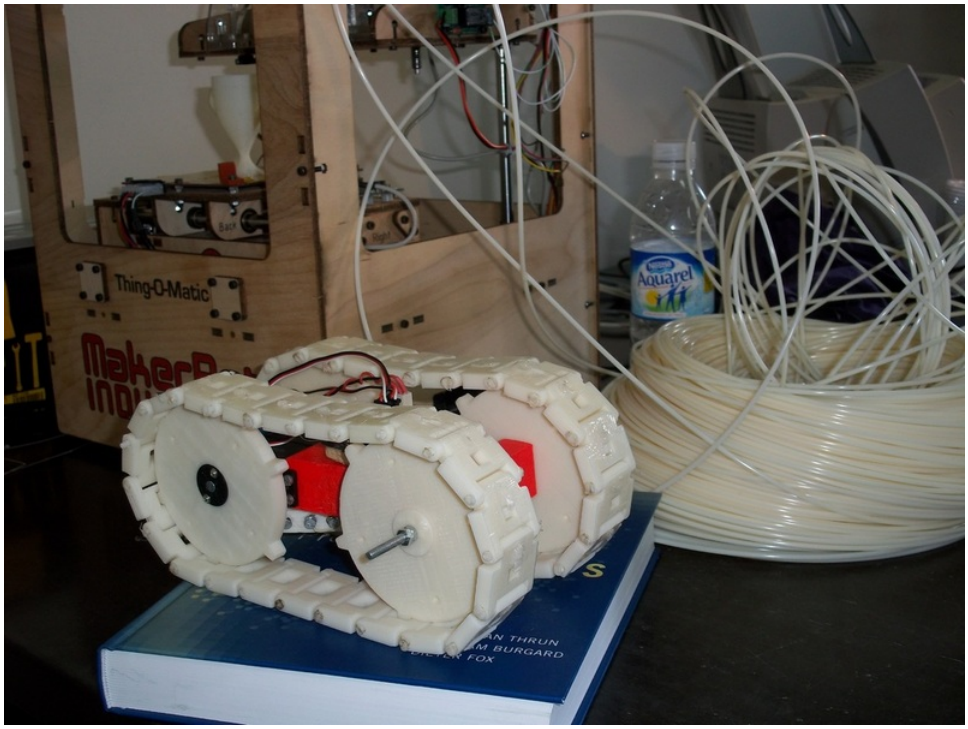

Figura 7.2: Orugator

Adaptando las ruedas y/o las orugas para posibilitar que engranen, la adición de unas orugas como éstas o similares puede resultar una idea interesante de cara a comparar ambos tipos de sistema de tracción y transmisión.

7.4. Mejoras electrónicas

7.4.1. Adición de una brújula

Con el acelerómetro, bajo determinadas suposiciones, es posible determinar el ángulo de cabeceo y de alabeo del robot. Sin embargo, el ángulo de guiñada no se puede calcular por los motivos explicados en la sección 4.3.2. La adición de una brújula permitiría calcular dicho ángulo de forma sencilla. La figura 7.3 muestra un ejemplo de brújula compuesta por un módulo HMC6352, con interfaz I2C, alimentable a las tensiones ya disponibles en nuestro circuito y con una precisión de 0.5°

Figura 7.3: Brújula HMC6352

7.4.2. Creación de una IMU completa

En nuestro proyecto ya está instalado un acelerómetro. Si añadimos un giroscopio, es posible crear un sistema de navegación inercial completo, ya que con ambos sensores es posible determinar en todo momento la orientación y la aceleración del vehículo. Como alternativa, se puede sustituir el acelerómetro por una IMU completa, si bien el autor considera que la construcción de la IMU a partir de acelerómetro y giroscopio resulta un ejercicio más interesante.

Por último, es posible imitar el esquema de las IMUs empleadas en aviación añadiendo un magnetómetro y programando un filtro de Kalman para fusión sensorial, combinando los datos de la IMU y el magnetómetro para obtener una señal de mejor calidad que cada una de las medidas por separado.

7.4.3. Adición de un módulo y antena GPS

Especialmente de cara a la profundización del comportamiento autónomo del robot, una propuesta bastante interesante consiste en la adición de un módulo GPS con la antena. La obtención de valores de posición mediante el GPS permite desarrollar algoritmos de navegación autónoma más complejos y corregir el error en posición de la IMU.

7.4.4. Montaje de los sensores de infrarrojos en un miniservo

Para mejorar las capacidades de percepción del robot, se propone sustituir el soporte para el sensor por un soporte para un miniservo que gire el sensor de infrarrojos a diferentes posiciones, de modo que el robot pueda percibir la distancia al terreno en más direcciones.

7.4.5. Control en velocidad

Actualmente, el control de los servomotores modificados queda en lazo abierto, por lo que la velocidad en estos puede ser inferior al valor deseado una vez entran en juego pares resistentes. Una idea útil sería el diseño de un esquema de control en lazo cerrado, lo que requeriría instalar un sensor para medir la velocidad del motor.

7.4.6. Control mediante un gamepad

Otra propuesta, destinada en este caso a crear un mando teleoperado en lugar de un comportamiento autónomo, consistiría en emplear un gamepad para controlar el robot a distancia. Esta idea requiere no sólo diseñar la electrónica que hace interfaz con el gamepad, sino seleccionar, diseñar e implementar un sistema de comunicación inalámbrica. para comunicar el robot con el dispositivo.

7.5. Mejoras al software

7.5.1. Implementación de las mejoras electrónicas

La adición de sensores o subsistemas electrónicos lleva aparejada una serie de cambios en el software. En esta subsección incluimos todos los cambios asociados a las mejoras propuestas en la sección 7.4

7.5.2. Creación de una interfaz gráfica de usuario para el programa de control

Una propuesta de mejora para el software del PC sería añadir una interfaz gráfica al programa que realice las misma funciones que la interfaz de texto implementada. Se propone el empleo de librerías como GTK+ o Qt, por ser software libre, multiplataforma y existir bindings para ambas librerías en Python.

7.5.3. Creación de más comportamientos autónomos

Se han implementado dos comportamientos autónomos: un movimiento de ascenso a un obstáculo y un movimiento de descenso, que demuestran la funcionalidad básica del vehículo. Una forma de enriquecer las capacidades del vehículo sería continuar añadiendo comportamientos autónomos para un mayor número de situaciones (por ejemplo, la superación de obstáculos con diferente altura a diferentes lados del robot)

7.5.4. Diseño de un algoritmo autónomo de navegación

Si el sistema de percepción del robot se amplía lo suficiente, es posible crear un robot totalmente autónomo que navegue buscando completar un criterio (por ejemplo, encontrar un objeto o alcanzar una posición). La navegación debería tener en cuenta los obstáculos y muros del terreno y decidir si son superables o si deberían ser esquivados.

Capítulo 8

Anexos

Lista de anexos:

- El anexo 8.1 es el presupuesto
- El anexo 8.2 es la planificación

8.1. Presupuesto

El coste unitario del robot se resume en la tabla 8.1

Objeto	#	Precio unitario	Total
Tornillería (véase tabla 6.2)	1	10 €	10 €
Rollo de ABS	1 Kg	20 €/Kg	20 €
Arduino Mega2560	1	45 €	45 €
Sharp GP2D120XJ00F	2	12 €	24 €
ADXL335	1	24 €	24 €
Componentes electrónicos*	1	10 €	10 €
Servos Futaba S3003	4	8 €	32 €
Servos Futaba S3010	4	20 €	80 €
Correas de goma 3mm×120mm	8	2 €	16 €
Acetona	1 l	2.5 €/l	2.5 €
Pegamento de plásticos	1	6 €	6 €
Total:			269.5 €

Cuadro 8.1: Presupuesto

Nota: “componentes eletrónicos” incluye cable, estaño, placa perforada, pines, un LM317, resitencias y condensadores.

Nótese que se ha realizado una estimación del coste en la que se omite el coste de la mano de obra, puesto que se supone que la proporciona el propio usuario. Dado que se pretende estimar el coste para un tercero que busque crear su propia copia del robot, tampoco se han contabilizado los costes de diseño (fundamentalmente, la mano de obra; se han empleado herramientas libres sin costes de licencia asociados)

8.2. Planificación

La figura 8.1 muestra el diagrama de Gantt del proyecto, mostrando en qué periodos se han llevado a cabo las diferentes fases del proyecto.

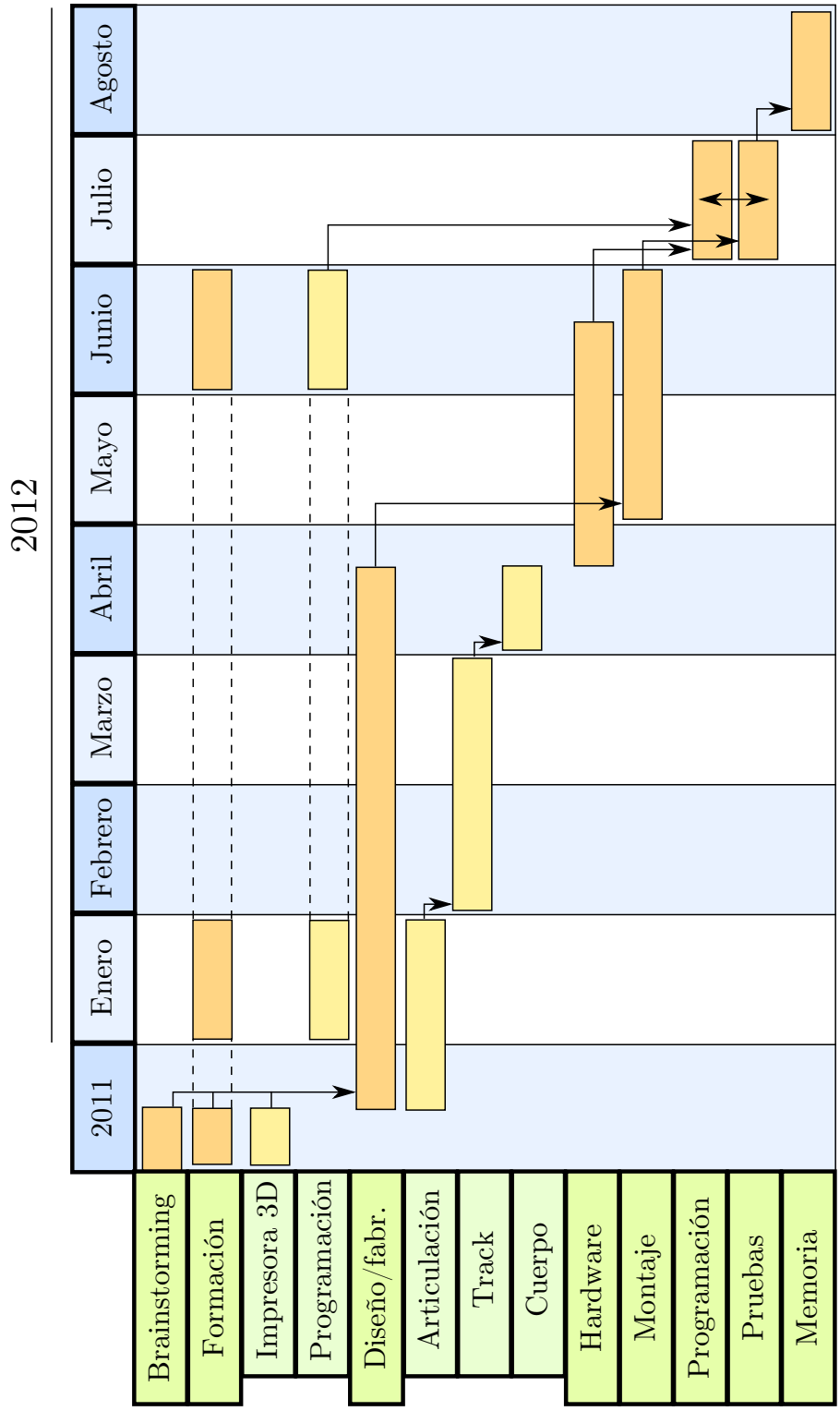

Figura 8.1: Diagrama de Gantt

Capítulo 9

Bibliografía

[1] Juan González, Alberto Valero, Andrés Prieto, Mohamed Abderrahim, "A New Open Source 3D-printable Mobile Robotic Platform for Education" (2012), *Advances in Autonomous Mini Robots*, Springer Books

[2] S Bradshaw, A Bowyer and P Haufe, "The Intellectual Property Implications of Low-Cost 3D Printing", (2010), *ScriptEd*, 7 (1), pp. 5-31.

[3] Jon Goitia, "4Track" `http://www.thingiverse.com/thing:13298` (23 de agosto de 2012)

[4] Juan González Gómez, "MiniSkyBot 1.0" `http://www.thingiverse.com/thing:7989` (23-8-2012)

[5] François Michaud et al., "Multi-Modal Locomotion Robotic Platform Using Leg-Track-Wheel Articulations" (2005), *Autonomous Robots* 18, pp. 137–156

[6] Richard M. Stallman, "The GNU Project" `http://www.gnu.org/gnu/thegnuproject.html` (23-8-2012)

[7] "What is Free Software?" `http://www.gnu.org/philosophy/free-sw.html` (23-8-2012)

[8] "The Open Source Definition" `http://opensource.org/docs/osd` (23-8-2012)

[9] Iván González, Juan González, Francisco Gómez-Arribas, "Hardware libre: clasificación y desarrollo de hardware reconfigurable en entornos GNU/Linux" (2003) `http://www.iearobotics.com/personal/juan/publicaciones/art4/hardware-libre.pdf`

[10] "RepRap", `http://www.reprap.org/wiki/RepRap` (23-8-2012)

[11] "Arduino", `http://arduino.cc/`

[12] "ABS - acrylonitrile butadiene styrene", `http://designinsite.dk/htmsider/m0007.htm` (24-8-2012)

[13] "RepRap Wiki - CAM Toolchains", `http://www.reprap.org/wiki/CAM_Toolchains` (24-8-2012)

[14] Beer, Johnston, *Mecánica vectorial para ingenieros* (1990) Mc. Graw-Hill.

[15] Paul Tipler, *Física para la ciencia y la tecnología, Vol. I* (2005) Ed. reverté 2005.

[16] "HobbyKing UBEC 3A / 2-6s LiPO", `http://www.hobbyking.com/hobbyking/store/__14310__HobbyKing_UBEC_3A_2_6s_LiPO.html` (25-8-2012)

[17] "LM317 - 3-Terminal Positive Adjustable Regulador" `http://www.fairchildsemi.com/ds/LM/LM317.pdf`

[18] “Sharp GP2D120XJ00F”, https://www.sparkfun.com/datasheets/Sensors/Infrared/GP2D120XJ00F_SS.pdf

[19] “ADXL335”, vhttp://www.sparkfun.com/datasheets/Components/SMD/adxl335.pdf

[20] “Arduino Mega2560”, http://arduino.cc/en/Main/ArduinoBoardMega2560 (26-8-2012)

[21] “Servos Futaba 3003”, http://www.iearobotics.com/wiki/index.php?title=Servos_Futaba_3003 (26-8-2012)

[22] “Language Reference”, http://arduino.cc/en/Reference/HomePage (27-8-2012)

[23] “pySerial API”, http://pyserial.sourceforge.net/pyserial_api.html (27-8-2012)

[24] “curses - Terminal handling for character-cell displays”, http://docs.python.org/library/curses.html(28-8-2012)

[25] Olalla Bravo, “Caterpillator”, http://www.thingiverse.com/thing:8559 (28-8-2012)

[26] “RepRap Wiki - Printing Material Supplies” http://reprap.org/wiki/Printing_Material_Suppliers